Comments Welcome

Lecture Notes On Solution Methods for Microeconomic Dynamic Stochastic Optimization Problems

Christopher D. Carroll[†]
ccarroll@jhu.edu

March, 1999

Abstract
These lecture notes sketch a set of techniques that are useful in solving microeconomic dynamic stochastic optimization problems. I make no attempt at a systematic overview of the many possible technical choices; instead, the notes present a very specific set of methods and techniques that I have found useful in the past. Associated with these notes is a set of Mathematica programs that solve the problems described in the text. Both text and programs are available on my home page, http://www.econ.jhu.edu/People/Carroll. These notes were originally written for my Advanced Topics in Macroeconomic Theory class at Johns Hopkins University.

The only compensation I request for the use of these programs and notes is that you cite the papers for which I originally developed the programs. Those papers are:

"Buffer Stock Saving: Some Macroeconomic Evidence," *Brookings Papers on Economic Activity, 1992:2*, pp. 61–156

"Buffer Stock Saving and the Life Cycle/Permanent Income Hypothesis," *Quarterly Journal of Economics*, CXII(1), February 1997, pp. 1–56

"Unemployment Expectations, Jumping (S,s) Triggers, and Household Balance Sheets," *NBER Macroeconomics Annual, 1997*, ed. Benjamin S. Bernanke and Julio Rotemberg, MIT Press.

[†]Department of Economics, Johns Hopkins University, Baltimore, MD 21218-2685, 410-516-7602 (office), 410-516-7600 (fax).

1 Introduction

These notes describe the solution of several sample dynamic stochastic optimization problems using *Mathematica.* The first problem solved is a consumption/saving problem similar to the problem considered in Carroll (1997), while the second problem solved is a two-state-variable consumption/saving problem similar to that in Carroll, Overland, and Weil (1998), where the second state variable is the stock of 'habits' that the consumer is used to satisfying. The third problem adds portfolio choice between a safe and a risky asset to the first problem, and shows how to solve this multiple-control problem. The tricks and techniques used in solving these problems have broad applicability to many dynamic stochastic optimization problems.

2 The Problem

Consider the following standard dynamic programming problem faced by a finite-lifetime consumer. The consumer's goal is to

$$\begin{aligned}
&\max && E_t \sum_{s=t}^{T} \beta^{s-t} u(C_s) && (1)\\
\text{s.t.} \quad & S_s && = X_s - C_s && (2)\\
& X_{s+1} && = R_{s+1} S_s + Y_{s+1} && (3)\\
& Y_{s+1} && = P_{s+1} \epsilon_{s+1} && (4)
\end{aligned}$$

where the variables are

β	–	time discount factor
X_s	–	resources available for consumption ('cash-on-hand')
S_s	–	savings in period s (portion of resources X_t not consumed)
C_s	–	consumption in period s
R_s	–	gross interest rate from period $s-1$ to s
$u(C)$	–	utility derived from consumption
Y_s	–	noncapital income in period s
P_s	–	'permanent income' in period s

The exogenous variables in the program evolve as follows:

$$\begin{aligned}
R_s &= R \text{ (a constant interest rate)}\\
P_{s+1} &= G_{s+1} P_s\\
\log\ \epsilon &\sim N(-\sigma^2/2, \sigma^2)
\end{aligned}$$

This last assumption guarantees that $\log E[\epsilon] = 0$ which means that $E[\epsilon] = 1$, and we are assuming that the average profile of income growth over the lifetime G_t is nonstochastic. Finally, assume that the utility function is of the CRRA form, $u(C) = C^{1-\rho}/(1-\rho)$.

3 Renormalization

The first, and probably the single most important, method for simplifying problems of this type is to see if you can redefine the problem in order to reduce the number of state variables. In this case the obvious idea is to see whether the problem can be rewritten in terms of the ratio of various variables to permanent income.

Consider the problem in the last period of life. The optimal plan in the last period of life is to consume everything. Designate the value function which yields total expected discounted utility from behaving optimally now and forevermore as $\mathcal{V}_t(X_t, P_t)$. Then we know that

$$\mathcal{V}_T(X_T, P_T) \quad = \quad \frac{X_T^{1-\rho}}{1-\rho}. \tag{5}$$

Now define small-case variables as the upper-case variable divided by the level of permanent income in the same period, so that, for example, $x_T = X_T/P_T$. Then equation (5) can be rewritten as

$$\mathcal{V}_T(X_T, P_T) \quad = \quad P_T^{1-\rho}\frac{x_T^{1-\rho}}{1-\rho}. \tag{6}$$

and define $V_T(x_T) = \frac{x_T^{1-\rho}}{1-\rho} = \mathcal{V}_T(X_T, P_T)/P_T^{1-\rho}$. Now note that it is possible to rewrite the accumulation equation for X_T as:

$$\begin{aligned}
X_T/P_T &= R[X_{T-1} - C_{T-1}]/P_T + Y_T/P_T & (7)\\
x_T &= R[\frac{X_{T-1}}{P_{T-1}} - \frac{C_{T-1}}{P_{T-1}}]\frac{P_{T-1}}{P_T} + \frac{Y_T}{P_T} & (8)\\
x_T &= \frac{R}{G_T}[x_{T-1} - c_{T-1}] + \epsilon_T & (9)
\end{aligned}$$

Now consider the problem in period $T-1$:

$$\begin{aligned}
\mathcal{V}_{T-1}(X_{T-1}, P_{T-1}) = & \quad \max_{\{C_{T-1}\}} u(C_{T-1}) + \beta E_{T-1}[\mathcal{V}_T(X_T, P_T)] & (10)\\
& \text{s.t.} \quad S_{T-1} = X_{T-1} - C_{T-1} & (11)\\
& \qquad X_T = RS_{T-1} + Y_T & (12)
\end{aligned}$$

Substituting in the constraints and equation (6) and designating the optimal level of period T-1 consumption as C^*_{T-1} and the optimal consumption/permanent income ratio as c^*_{T-1}

$$\begin{aligned}
\mathcal{V}_{T-1}(X_{T-1}, P_{T-1}) &= \frac{(C^*_{T-1})^{1-\rho}}{1-\rho} + \beta E_{T-1}[P_T^{1-\rho}\frac{x_T^{1-\rho}}{1-\rho}] & (13)\\
&= P_{T-1}^{1-\rho}\frac{(c^*_{T-1})^{1-\rho}}{1-\rho} + (P_{T-1}G_T)^{1-\rho}\beta E_{T-1}\left[\frac{x_T^{1-\rho}}{1-\rho}\right] & (14)\\
&= P_{T-1}^{1-\rho}\left[\frac{(c^*_{T-1})^{1-\rho}}{1-\rho} + G_T^{1-\rho}\beta E_{T-1}\left[\frac{x_T^{1-\rho}}{1-\rho}\right]\right] & (15)\\
&= P_{T-1}^{1-\rho}\left[\frac{(c^*_{T-1})^{1-\rho}}{1-\rho} + G_T^{1-\rho}\beta E_{T-1}[V_T(x_T)]\right] & (16)
\end{aligned}$$

or, if we define $V_{T-1}(x_{T-1}) = \mathcal{V}_{T-1}(X_{T-1}, P_{T-1})/P_{T-1}^{1-\rho}$ this is

$$V_{T-1}(x_{T-1}) \quad = \quad \left[\frac{(c^*_{T-1})^{1-\rho}}{1-\rho} + G_T^{1-\rho}\beta E_{T-1}[V_T(x_T)]\right] \tag{17}$$

Analogously,

$$\begin{aligned}
\mathcal{V}_{T-2}(X_{T-2},P_{T-2}) &= \frac{(C^*_{T-2})^{1-\rho}}{1-\rho} + \beta E_{T-2}[\mathcal{V}_{T-1}(X_{T-1},P_{T-1})] && (18)\\
&= P_{T-2}^{1-\rho}\frac{(c^*_{T-2})^{1-\rho}}{1-\rho} + \beta E_{T-2}P_{T-1}^{1-\rho}[V_{T-1}(x_{T-1})] && (19)\\
&= P_{T-2}^{1-\rho}\frac{(c^*_{T-2})^{1-\rho}}{1-\rho} + (G_{T-1}P_{T-2})^{1-\rho}\beta E_{T-2}[V_{T-1}(x_{T-1})] && (20)\\
&= P_{T-2}^{1-\rho}\left[\frac{(c^*_{T-2})^{1-\rho}}{1-\rho} + G_{T-1}^{1-\rho}\beta E_{T-2}[V_{T-1}(x_{T-1})]\right] && (21)
\end{aligned}$$

or

$$V_{T-2}(x_{T-2}) = \left[\frac{(c^*_{T-2})^{1-\rho}}{1-\rho} + G_{T-1}^{1-\rho}\beta E_{T-2}[V_{T-1}(x_{T-1})]\right] \quad (22)$$

The exact same logic can be repeated back an arbitrary number of periods. Hence if one solves the maximization problem

$$\begin{aligned}
V_t(x_t) &= \max_{\{c_t\}} u(c_t) + G_{t+1}^{1-\rho}\beta E_t[V_{t+1}(x_{t+1})] && (23)\\
&\text{such that}\\
x_{t+1} &= \left(\frac{R}{G_{t+1}}\right)s_t + \epsilon_{t+1}\\
s_t &= x_t - c_t
\end{aligned}$$

which has only a single state variable, one can obtain the levels of the value function, consumption, and all other variables of interest simply by multiplying the results from this optimization problem by the appropriate factor of P_t. Hence we have reduced a problem in two state variables to a single-state-variable problem.

Henceforth we will take the single-state-variable problem defined in (23) as the problem under consideration, and to simplify matters further we will assume that permanent income remains constant $G_t = 1 \ \forall\ t$.[1]

4 The Usual Theory, and Some Notation

The usual theoretical analysis of this problem proceeds as follows.

Because the optimal plan in period T is to consume everything, and there is no utility beyond the last period of life, we have

$$\begin{aligned}
V_T(x_T) &= u(c_T(x_T)) && (24)\\
&= u(x_T) && (25)
\end{aligned}$$

Then in the second-to-last period of life the problem can be rewritten

$$\begin{aligned}
V_{T-1}(x_{T-1}) = & && (26)\\
\max_{\{c_{T-1}\}} \quad u(c_{T-1}) \quad & +\beta E_{T-1}[V_T(x_T)] && (27)
\end{aligned}$$

[1]Allowing for growth in permanent income adds no substantive difficulty to the problem, and allowing permanent income to be subject to stochastic shocks also is relatively straightforward.

and more generally for any period t earlier than T the problem is

$$V_t(x_t) = \tag{28}$$
$$\max_{\{c_t\}} \quad u(c_t)+ \quad \beta E_t[V_{t+1}(x_{t+1})] \tag{29}$$
$$\text{such that}$$
$$x_{t+1} = R[x_t - c_t] + \tilde{\epsilon}_{t+1} \tag{30}$$

the first order condition for which is

$$u'(c_t) = \beta E_t[RV'_{t+1}(x_{t+1})] \tag{31}$$

and because the Envelope theorem tells us that

$$V'_t(x_t) = \beta E_t[RV'_{t+1}(x_{t+1})] \tag{32}$$

we can substitute this expression for the RHS of equation (31) to get

$$u'(c_t) = V'_t(x_t) \tag{33}$$

and rolling this equation forward one period

$$u'(c_{t+1}) = V'_{t+1}(x_{t+1}) \tag{34}$$

and substituting in equation (31) gives us the Euler equation for consumption

$$u'(c_t) = \beta E_t[Ru'(c_{t+1})]. \tag{35}$$

Now substitute the budget constraint into the maximization problem:

$$V_t(x_t) = \tag{36}$$
$$\max_{\{c_t\}} \quad u(c_t)+ \quad \beta E_t[V_{t+1}(R[x_t - c_t] + \tilde{\epsilon}_{t+1})] \tag{37}$$

and note that neither x_t nor c_t has any *direct* effect on $E_t[V_{t+1}]$ - it is only the difference between them (i.e. unconsumed wealth or 'savings') that matters. It will be convenient to define a function

$$\Omega_t(s_t) = E_t[V_{t+1}(Rs_t + \tilde{\epsilon}_{t+1})] \tag{38}$$

which returns the expected value associated with any given amount of savings. Note also that this definition implies that

$$\Omega^{'}_t(s_t) = E_t[RV^{'}_{t+1}(Rs_t + \tilde{\epsilon}_{t+1})]. \tag{39}$$

or, substituting from equation (34),

$$\Omega^{'}_t(s_t) = E_t[Ru'(c_{t+1}[Rs_t + \tilde{\epsilon}_{t+1}])]. \tag{40}$$

Finally, note that the first order condition (31) can be rewritten as

$$u'(c_t) = \beta\Omega^{'}_t(x_t - c_t). \tag{41}$$

5 Solving the Next-To-Last Period

Consider again the second-to-last period of life. We have

$$V_{T-1}(x_{T-1}) = \max_{\{c_{T-1}\}} u(c_{T-1}) + \beta E_{T-1}[V_T(\tilde{x}_T)] \tag{42}$$

$$\text{s.t.} \quad s_{T-1} = x_{T-1} - c_{T-1} \tag{43}$$

$$x_T = R s_{T-1} + \epsilon_T \tag{44}$$

Substituting from the budget constraint and the definition of $u(c)$ this becomes:

$$V_{T-1}(x_{T-1}) = \max_{\{c_{T-1}\}} \frac{c_{T-1}^{1-\rho}}{1-\rho} + \beta E_{T-1}\left[\frac{(R[x_{T-1} - c_{T-1}] + \tilde{\epsilon}_T)^{1-\rho}}{1-\rho}\right] \tag{45}$$

and substituting the definition of the expectations operator:

$$V_{T-1}(x_{T-1}) = \max_{\{c_{T-1}\}} \frac{c_{T-1}^{1-\rho}}{1-\rho} + \beta \int_0^\infty \frac{(R[x_{T-1} - c_{T-1}] + \epsilon)^{1-\rho}}{1-\rho} dF(\epsilon) \tag{46}$$

In principle, this function implicitly defines a function $c_{T-1}(x_{T-1})$ which yields the optimal value of consumption in period $T-1$ for any given level of resources x_{T-1}. Unfortunately, however, there is no analytical solution to this maximization problem, and so for any given value of x_{T-1} we must use numerical routines to find the c_{T-1} that maximizes the expression. But this is excruciatingly slow because for every potential c_{T-1} to be considered, the integral must be calculated numerically, and numerical integration is *very* slow.

5.1 Discretizing the Distribution

The first time-saving step is therefore to construct a discrete approximation to the lognormal distribution that can be used rather than full-fledged numerical integration. An n-point approximation is calculated as follows.

Define n points on the $[0,1]$ interval as $\theta = [0, 1/(n-1), 2/(n-1), \ldots, 1]$. Call the inverse of the lognormal distribution $F_\epsilon^{-1}(p)$, and define the points $\theta_i^{-1} = F^{-1}(\theta_i)$. Then define

$$\epsilon_i = \int_{\theta_{i-1}^{-1}}^{\theta_i^{-1}} \epsilon \, dF(\epsilon) \tag{47}$$

The ϵ_i represent the mean values of ϵ in each of the regions bounded by the θ_i^{-1} endpoints. The method is illustrated in figure 1. The curve represents the CDF of $F_\epsilon(\epsilon)$, a lognormal distribution such that $E\epsilon = 1$ and $\sigma = .1$, and the dots represent the n equiprobable values of ϵ_i which are used to approximate this distribution.

There are more sophisticated methods available (most notably Gauss-Hermite Quadrature), but my experience is that this method is easy to understand, quick to calculate, and performs almost as well.

The maximization problem can now be rewritten

$$V_{T-1}(x_{T-1}) = \max_{\{c_{T-1}\}} \frac{c_{T-1}^{1-\rho}}{1-\rho} + \beta \frac{1}{n} \sum_{i=1}^{n} \frac{(R[x_{T-1} - c_{T-1}] + \epsilon_i)^{1-\rho}}{1-\rho} \tag{48}$$

or, recalling our definition of $\Omega_t(s_t)$ as the expected value of saving amount s_t in period t,

$$V_{T-1}(x_{T-1}) = \max_{\{c_{T-1}\}} \frac{c_{T-1}^{1-\rho}}{1-\rho} + \beta \Omega_{T-1}(x_{T-1} - c_{T-1}). \tag{49}$$

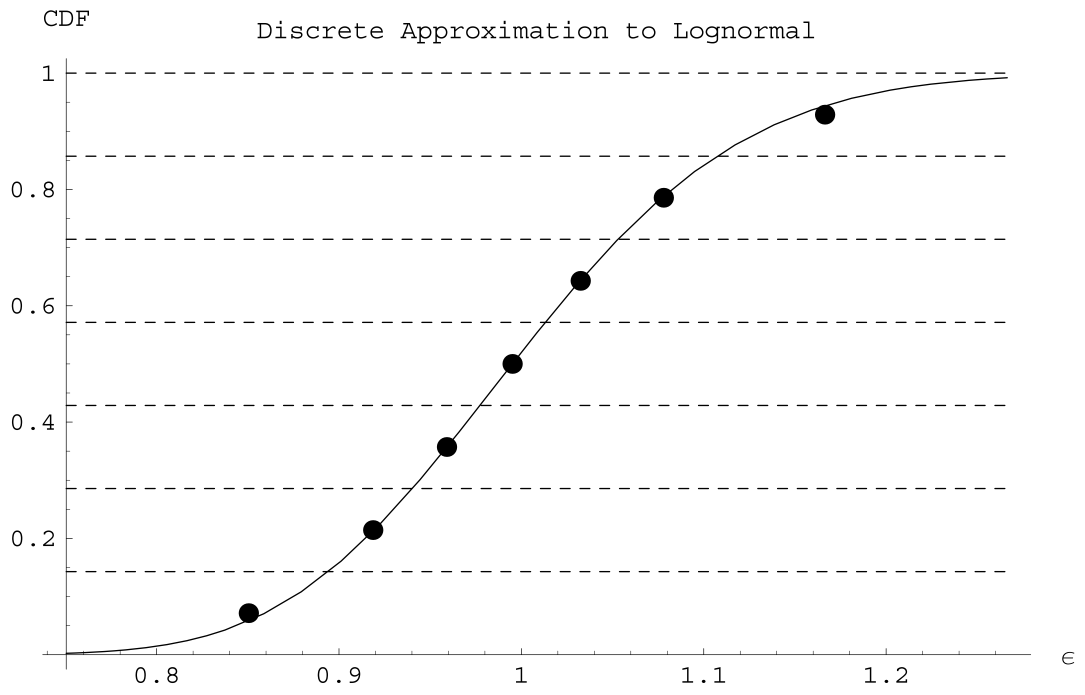

Figure 1:

5.2 The Approximate Consumption Function and the Approximate Value Function

Given a particular value of x_{T-1}, a numerical maximization routine can now be expected to find the c_{T-1} that maximizes this expression in a reasonable amount of time. This is done in the program `2period.m`.

The structure of the program is as follows. `setup_workspace.m` reads into memory a variety of useful functions and procedures that will be useful in solving all of the problems we will see. `setup_params.m` reads in values for parameters like the coefficient of relative risk aversion and the time preference rate. `setup_shocks.m` calculates the values for the ϵ_i defined above (and puts those values, and the probability associated with each of them, in the variables `ShockVals` and `ShockProb`.) Finally, `setup_grids.m` constructs a list of potential values of cash-on-hand and puts them in the variable xGrid $= \{0, 1, 2, 3, 4, 5\}$.

Next, the program defines a function `OmegaTm1Raw[s_]` which is the exact implementation of (38): this function returns the expectation of the value of next period's value function for any given amount of saving in period $t - 1$, i.e. how much that amount of saving is 'worth' in terms of units of present discounted value.

The heart of the program is the next expression. This expression loops over the values of the variable xGrid, solving the maximization problem[2]

$$\max_{\{c\}} \quad \text{u[c] + beta OmegaTm1Raw[xGrid[[i]]-c]} \tag{50}$$

for each of the i values of xGrid (henceforth let's call these points $x_{i,T-1}$). The maximization routine returns two values: the maximized value, and the value of c which yields that maximized value. When the loop (the `Table` command) is finished, the variable `MaxResultsTm1` contains two lists, one with the values $V_{i,T-1}$ and the other with the consumption levels $c_{i,T-1}$ associated with the $x_{i,T-1}$.

Now we use the first of the really convenient built-in features of Mathematica. Given data of the form $\{\{x_1, y_1\}, \{x_2, y_2\} \ldots, \{x_3, y_3\} \ldots\}$ Mathematica can create an object called an `InterpolatingFunction` which, when applied to an input x will yield the value of y which corresponds to a linear interpolation of

[2]Actually, Mathematica has a built-in minimization function, so we find the minimum of the negative of this expression.

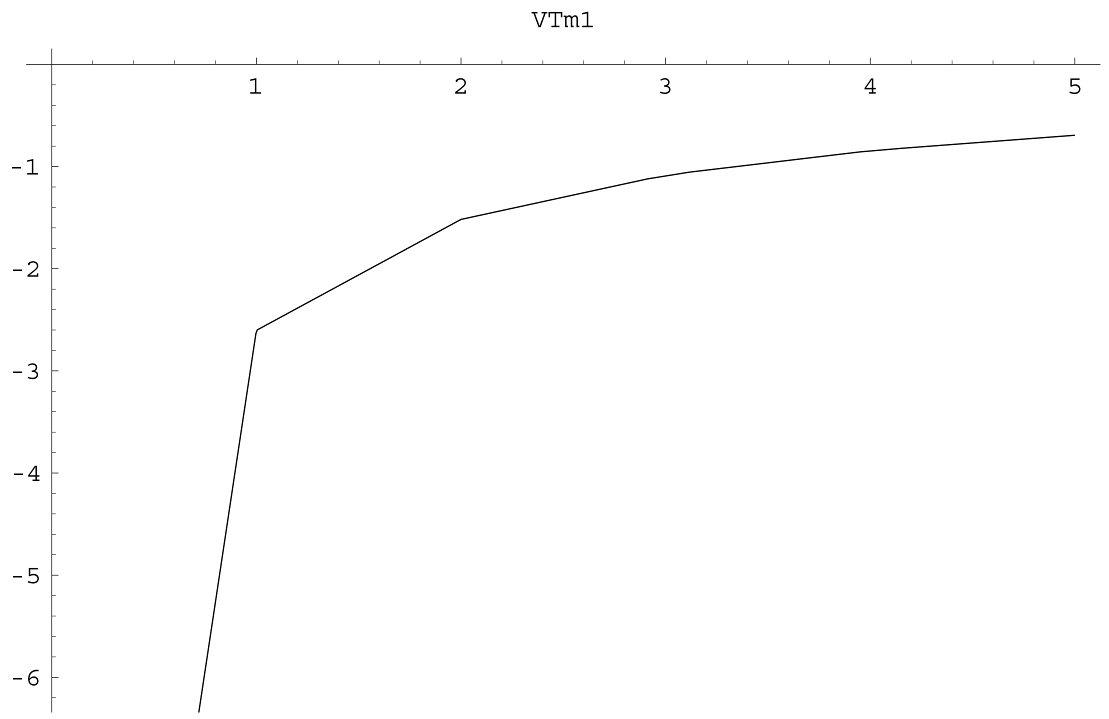

Figure 2: $\hat{V}_{T-1}(x_{T-1})$

the value of y from the points in the `InterpolatingFunction` object. We can therefore define a function $\hat{c}_{T-1}(x_{T-1})$ which, when called with an x_{T-1} that is equal to one of the points in xGrid$_i$ returns the associated value of $c_{i,T-1}$, and when called with a value of x_{T-1} that is not exactly equal to one of the xGrid$_i$, returns the value of C that represents a linear interpolation between the $c_{i,T-1}$ associated with the two xGrid$_i$ points nearest to x_{T-1}. Thus if the function is called with $x_{T-1} = 1.75$ and the nearest gridpoints are $x_{j,T-1} = 1$ and $x_{k,T-1} = 2$ then the value of c_{T-1} to be returned by the function would be $(.25c_{j,T-1} + .75c_{k,T-1})$. We can define a numerical approximation to the value function $\hat{V}_{T-1}(x_{T-1})$ in an exactly analogous way.

The figures 2 and 3 show plots of the cTm1 and VTm1 `InterpolatingFunctions` that are generated by the program. While the cTm1 function looks very smooth, the fact that the VTm1 function is a set of line segments is very obvious.

5.3 Interpolating Expectations

The program `2period.m` works fine in the sense that it generates a very good approximation to the true optimal consumption function. However, there is one obvious inefficiency in the program: For every value of x_{T-1}, the program must calculate the utility consequences of various possible choices of c_{T-1}. But for any given value of s_{T-1}, there is a good chance that the program may end up calculating the corresponding value many times coming from different x_{T-1}. For example, it is quite likely that the program will calculate the value of saving exactly zero dozens of times. It would be much more efficient if the program could make that calculation once and then merely recall the value when called upon.

This can be achieved using the same interpolation technique used above to construct a numerical value function: construct a grid of possible values for saving at time $T-1$, sGrid, designating the specific points $s_{i,T-1}$; and for each of these values of $s_{i,T-1}$, calculate $\Omega_{i,T-1} = \Omega_{T-1}(s_{i,T-1})$ using equation (38); and construct an `InterpolatingFunction` object $\hat{\Omega}_{T-1}(s_{T-1})$ from the list of values $\{\{s_{1,T-1}, \Omega_{1,T-1}\}, \{s_{2,T-1}, \Omega_{2,T-1}\} \ldots\}$.

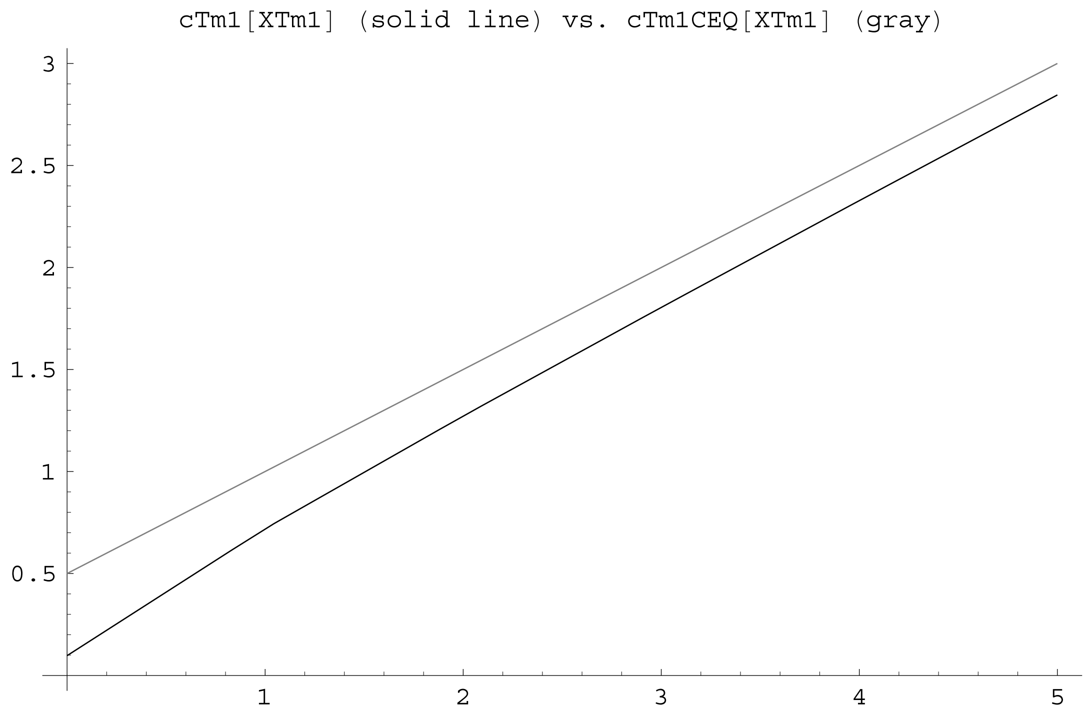

Figure 3: $c_{T-1}(x_{T-1})$

Thus, we are now interpolating for the expected value of saving; the program `2period_intexp.m` presents the results of such an experiment. Figure 4 compares the true value function to the `InterpolatingFunction` approximation; the two are of course identical at the gridpoints chosen for s_{T-1} and they appear reasonably close except in the region below $x_{T-1} = 1$.

Nevertheless, the results for the consumption function shown in figure 5 are surprisingly bad. For example, when x_{T-1} goes from 2 to 3, c_{T-1} goes from about 1 to about 2, yet when x_{T-1} goes from 3 to 4, c_{T-1} goes from about 2 to about 2.05. Thus the function fails even to be strictly concave, which is problematic because Carroll and Kimball (1996) prove that the consumption function is strictly concave in problems like this one.

5.4 Value Function Versus First Order Condition

Loosely speaking, the problem is that *behavior* is determined by the *marginal* value function, not by the level of the value function. To see this, recall that a quadratic utility function exhibits risk aversion because

$$E[-(\tilde{c} - c^*)^2] < -(E[\tilde{c}] - c^*)^2. \tag{51}$$

for c less than the 'bliss point' of c^* even though the consumption/saving *behavior* of consumers with quadratic utility is unaffected by risk. The reason behavior is unaffected by risk is that behavior is determined by the first order condition, which depends on *marginal* utility, and marginal utility is unaffected by risk:

$$E[-2(\tilde{c} - c^*)] = -2(E[\tilde{c}] - c^*) \tag{52}$$

Intuitively speaking, if one's goal is to accurately capture behavior that is governed by marginal utility or the marginal value function, numerical techniques that approximate the *marginal* value function are likely to lead to a more accurate approximation to optimal behavior than techniques that operate on the level of the value function.

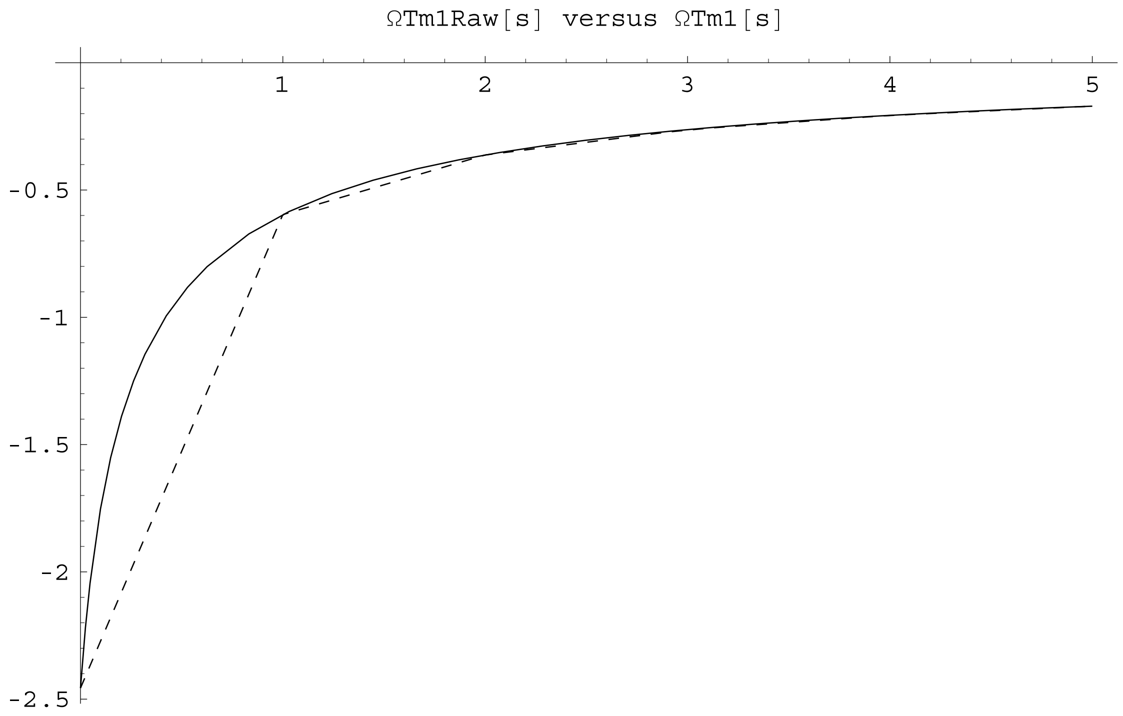

Figure 4:

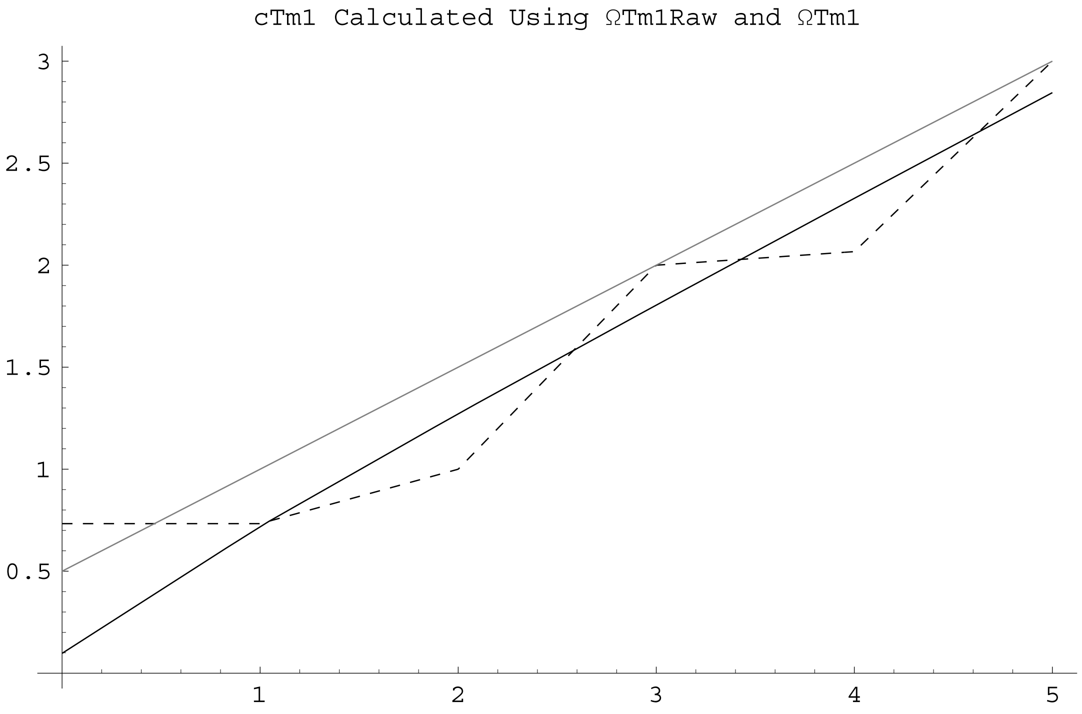

Figure 5:

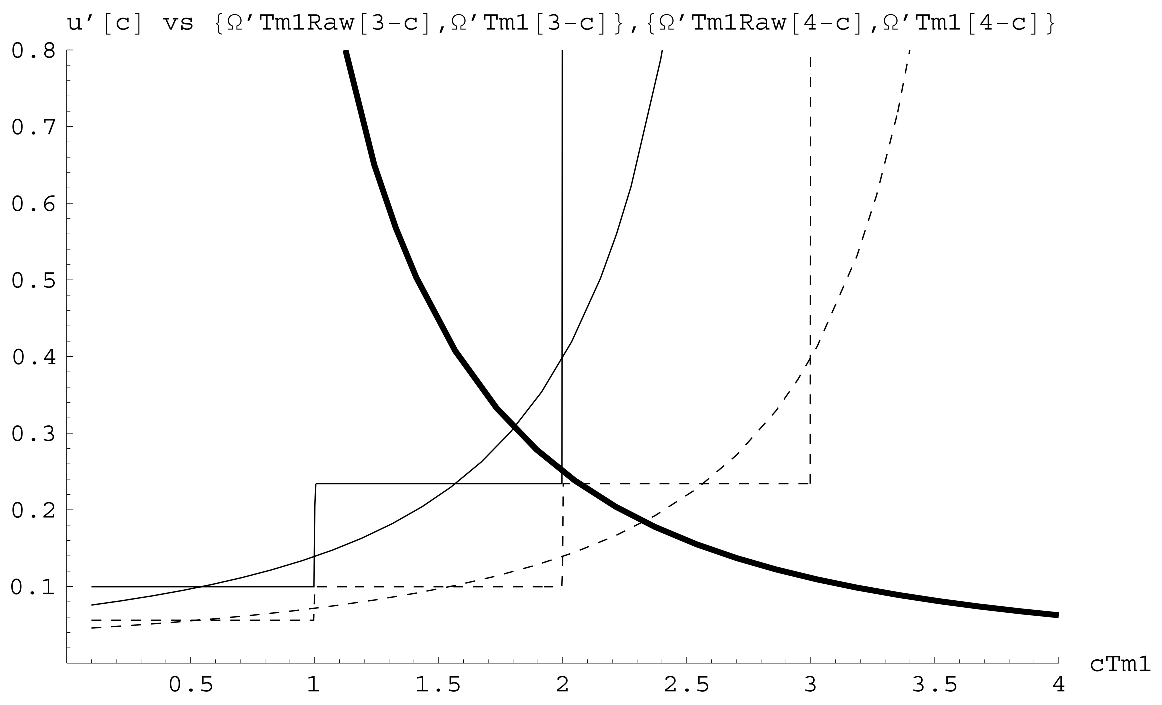

Figure 6:

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

$$u'(c_{T-1}) = \beta E_{T-1}[Ru'(c_T)] \tag{53}$$

$$c_{T-1}^{-\rho} = \beta R \left(\frac{1}{n}\right) \sum_{i=1}^{n} [R(x_{T-1} - c_{T-1}) + \epsilon_i]^{-\rho} \tag{54}$$

The downward-sloping curve in figure 6 shows the value of $c_{T-1}^{-\rho}$ for our baseline parameter values for $0 \leq c_{T-1} \leq 4$ (the horizontal axis). The solid upward-sloping curve shows the value of the RHS of (54) as a function of c_{T-1} under the assumption that $x_{T-1} = 3$. Constructing this figure is rather time-consuming, because for every value of c_{T-1} plotted we must calculate the RHS of (54). The value of c_{T-1} for which the RHS and LHS of (54) are equal is the optimal level of consumption given that $x_{T-1} = 3$, so the intersection of the downward- sloping and the upward-sloping curves gives the optimal value of c_{T-1}. As we can see, the two curves intersect just below $c_{T-1} = 2$. Similarly, the upward-sloping dashing curve shows the expected value of the RHS of (54) under the assumption that $x_{T-1} = 4$, and the intersection of this curve with $u'[c_{T-1}]$ yields the optimal level of consumption if $x_{T-1} = 4$. These two curves intersect slightly below $c_{T-1} = 2.5$. Thus, increasing x_{T-1} from 3 to 4 increases optimal consumption by about 0.5.

Now consider the derivative of our function $\hat{\Omega}_{T-1}(s_{T-1})$. Because we have constructed $\hat{\Omega}_{T-1}$ as a linear interpolation, the slope of $\hat{\Omega}_{T-1}(s_{T-1})$ between any two adjacent points $\{s_{i,T-1}, s_{i+1,T-1}\}$ is constant. The level of the slope immediately below any particular gridpoint is different, of course, from the slope above that gridpoint, a fact which implies that the derivative of $\hat{\Omega}_{T-1}(s_{T-1})$ follows a step function.

The solid-line step function in figure 6 depicts the actual value of $\hat{\Omega}'_{T-1}(3-c_{T-1})$. When we attempt to find optimal values of c_{T-1} given x_{T-1} using $\hat{\Omega}_{T-1}(s_{T-1})$, the numerical optimization routine will return the c_{T-1} for which $u'[c_{T-1}] = \hat{\Omega}_{T-1}(x_{T-1} - c_{T-1})$. Thus, for $x_{T-1} = 3$ the program will return the value of c_{T-1} for which the downward-sloping $u'[c_{T-1}]$ curve intersects with the $\hat{\Omega}'_{T-1}(3 - c_{T-1})$; as the diagram shows, this value is very close to 2. Similarly, if we ask the routine to find the optimal c_{T-1} for $x_{T-1} = 4$, it finds the point of intersection of $u'[c_{T-1}]$ with $\hat{\Omega}'_{T-1}(4 - c_{T-1})$; and as the diagram shows, this intersection

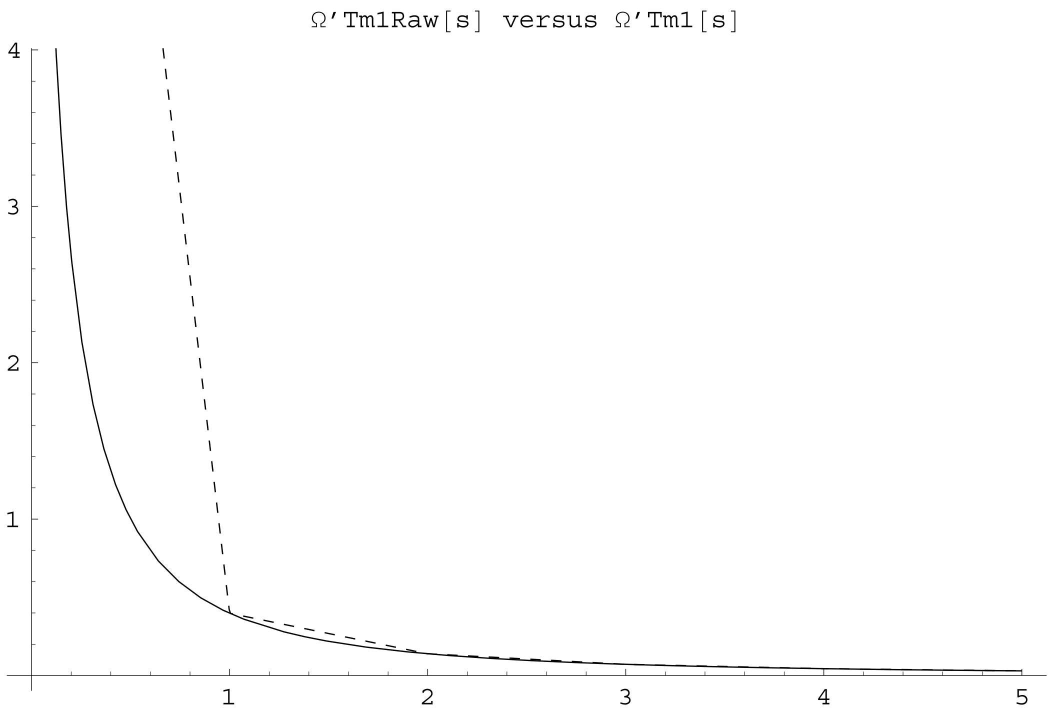

Figure 7:

is only slightly above 2. Hence, this figure shows why the numerical consumption function plotted earlier returned values very close to $c_{T-1} = 2$ for both $x_{T-1} = 3$ and $x_{T-1} = 4$.

Obviously, we would obtain much better estimates of the point of intersection between $u'(c_{T-1})$ and $\Omega'_{T-1}(x_{T-1} - c_{T-1})$ if our estimate of $\hat{\Omega}'_{T-1}$ were not a step function. In particular, we already know how to construct linear interpolations to functions, so the obvious idea to pursue now is to construct a linear interpolating approximation to the *marginal value of saving* function $\Omega^{'}(s_{T-1})$. That is, we calculate the value of

$$\Omega'_{T-1}(s_{T-1}) \quad = \quad R\left(\frac{1}{n}\right)\sum_{i=1}^{n}\left[Rs_{T-1} + \epsilon_i\right]^{-\rho} \tag{55}$$

at the points in sGrid yielding $\{\{s_{1,T-1}, \Omega_{1,T-1}\}, \{s_{2,T-1}, \Omega_{2,T-1}\}\ldots\}$ and construct $\hat{\Omega}'_{T-1}(s_{T-1})$ as the linear interpolating function that fits this set of points.

The program file `2period_intexp_foc.m` therefore defines a function `OmegaPrimeTm1Raw[s]` as the embodiment of equation (55), constructs the `InterpolatingFunction` as described above. The results are shown in figure 7. The linear interpolating approximation looks 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 much better approximation of the true consumption function (solid) than was the consumption function obtained by approximating the level of the value function (short dashes).

5.5 Transformation

However, for levels of x below 2, even the new function diverges noticeably from the optimum. That is because the linear interpolation does an increasingly poor job of capturing the nonlinearity of the $\Omega'_{T-1}(s_{T-1})$ function at lower and lower levels of s.

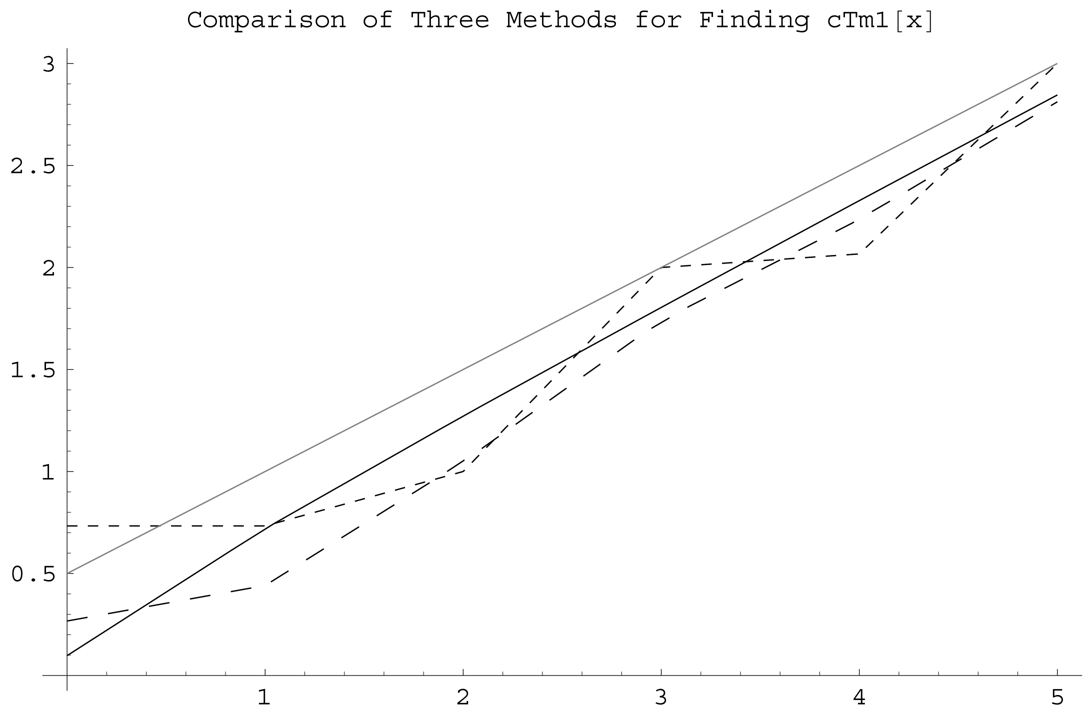

Figure 8:

This is where we bring our last trick into play. If $R = \beta = 1$ and there is no uncertainty (that is, we know for sure that income next period will be $E[\epsilon] = 1$) the Euler equation linking the marginal utility of consumption in period $T - 1$ to the marginal utility in period T is:

$$c_{T-1}^{-\rho} \quad = \quad c_T^{-\rho} \tag{56}$$

Now recall that in the case of the problem with no uncertainty and with $\beta = R = 1$, the optimal solution is to spend half of total lifetime resources in period $T - 1$ and the remainder in period T. Since total resources are known with certainty to be $x_{T-1} + 1$, and since we know that $V_{T-1}^{'}(x_{T-1}) = u'(c_{T-1})$ this implies that

$$V'_{T-1}(x_{T-1}) = (\frac{x_{T-1} + 1}{2})^{-\rho}$$

Note that for ρ much above 1 this becomes a highly nonlinear function. However, if we raise both sides of the equation to the power $(-1/\rho)$ it becomes a linear function:

$$[V'_{T-1}(x_{T-1})]^{-1/\rho} \quad = \quad \frac{x_{T-1} + 1}{2} \tag{57}$$

This is a specific example of a general phenomenon: Theoretical results cited in Carroll and Kimball (1996) establish 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 $(\alpha x_t)^{-\rho}$. This means that if we were solving the perfect certainty problem numerically, we could always calculate a numerically exact interpolation. Hence we define

$$\Lambda_t^{'}(x_t) \quad \equiv \quad [V_t^{'}(x_t)]^{-1/\rho} \tag{58}$$

(because Λ looks like an upside-down V and what we are doing is almost like taking the inverse) we can recover the value of $V_t(x_t)$ by calculating $[\Lambda_t^{'}(x_t)]^{-\rho}$. Similarly, define

$$\mho_t^{'}(s_t) \quad \equiv \quad [\Omega_t^{'}(s_t)]^{-1/\rho}. \tag{59}$$

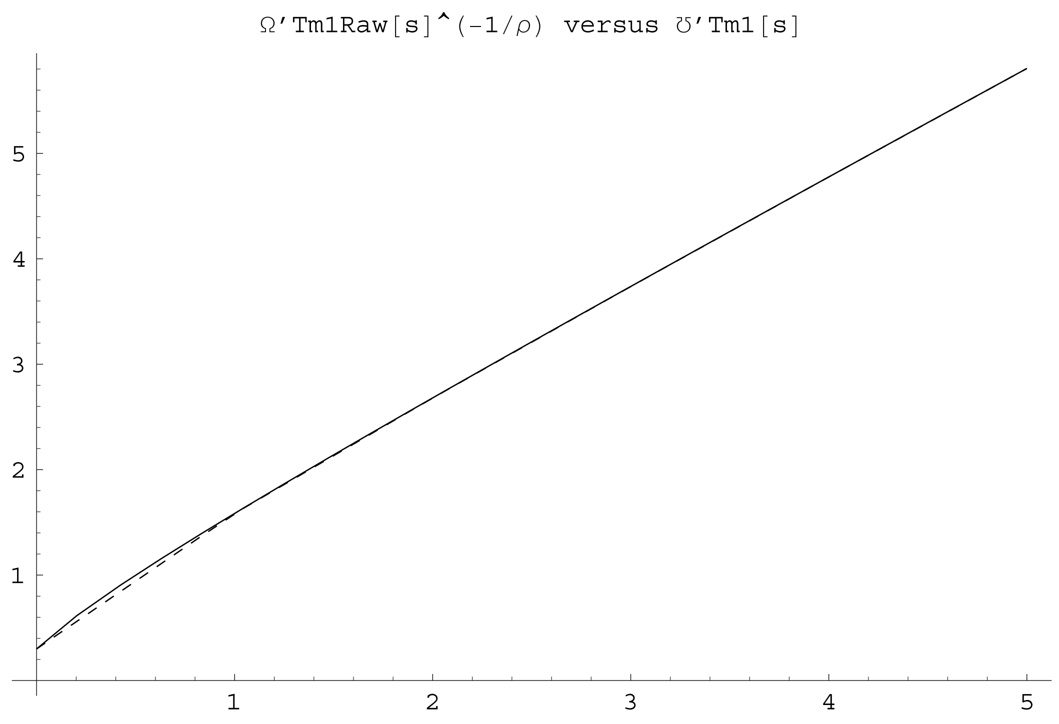

Figure 9:

To put this in intuitive terms, the problem we are facing is that the marginal utility function is highly nonlinear. But we have a great solution to that problem, because we know that much of that nonlinearity springs from the fact that we are raising something to the power $-\rho$. In effect, we can 'unwind' all of the nonlinearity owing to that parameter and the remaining nonlinearity will not be nearly so great. As illustration, look at figure 9 generated by the program which solves the optimization problem using these transformations, `2period_intexp_foc_inv.m`. (We use the text OmegaInv to stand for $\mho$ in the programs). The solid line calculates the exact numerical value of $\mho'_{T-1}(x_{T-1})$ while the dashing line is the linear interpolating approximation $\hat{\mho}'_{T-1}(x_{T-1})$. This figure illustrates the value of the transformation: the true figure is very close to linear, and so the linear approximation is almost indistinguishable from the true value except at the very lowest values of x_{T-1}. Figure 10 shows that when we calculate $\hat{\Omega}'_{T-1}(s_{T-1})$ as $[\hat{\mho}'_{T-1}(s_{T-1})]^{-1/\rho}$ (dashing line) we obtain a *much* closer approximation to the true function $\Omega'_{T-1}(s_{T-1})$ (solid line) than we did in the previous program which did not do the transformation. Since this is the precise function that will be used in finding the optimum via the FOC, this figure makes it clear why the transformation is so valuable.

Figure 11 shows that the consumption function that emerges from this program is now very close to the 'true' consumption function all the way down to $x_{T-1} = 0$. Although it is still true that you can see small deviations, the function is now being constructed using literally thousands of times fewer computations than would have been required without all these tricks. We could increase the density of the gridpoints for s_{T-1} by a factor of ten and still be able to solve the problem vastly more quickly than it can be solved without these techniques.

Note that the appropriate transformation for V_{T-1} is different from that for V'_{T-1}. The idea is to transform the object in such a way that, in a perfect-certainty world, the resultsing function would be linear. It turns out that in the perfect certainty world, the value function corresponding to a maximization problem with CRRA period utility functions is itself a CRRA function with the same parameter; that is, $V_t(x_t) = \frac{(\alpha x_t)^{1-\rho}}{1-\rho}$ for some α. The appropriate transformation is therefore $\Lambda_t(x_t) = ((1-\rho)V_t(x_t))^{1/(1-\rho)}$ and similarly $\mho_t(x_t) = ((1-\rho)\Omega_t(x_t))^{1/(1-\rho)}$.

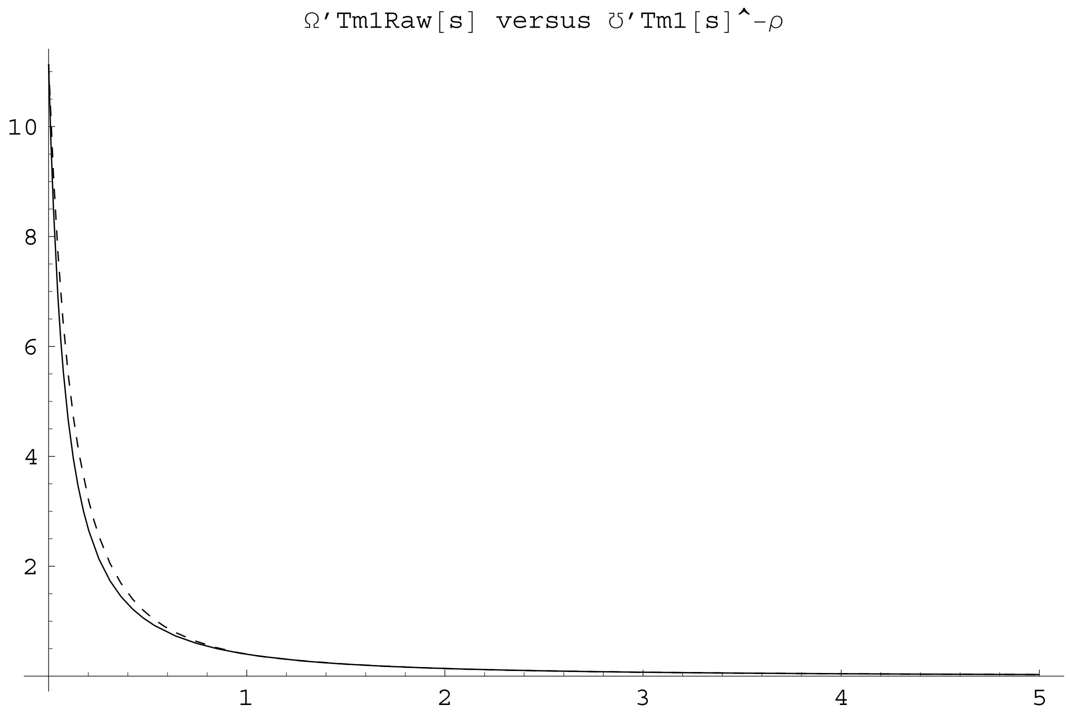

Figure 10:

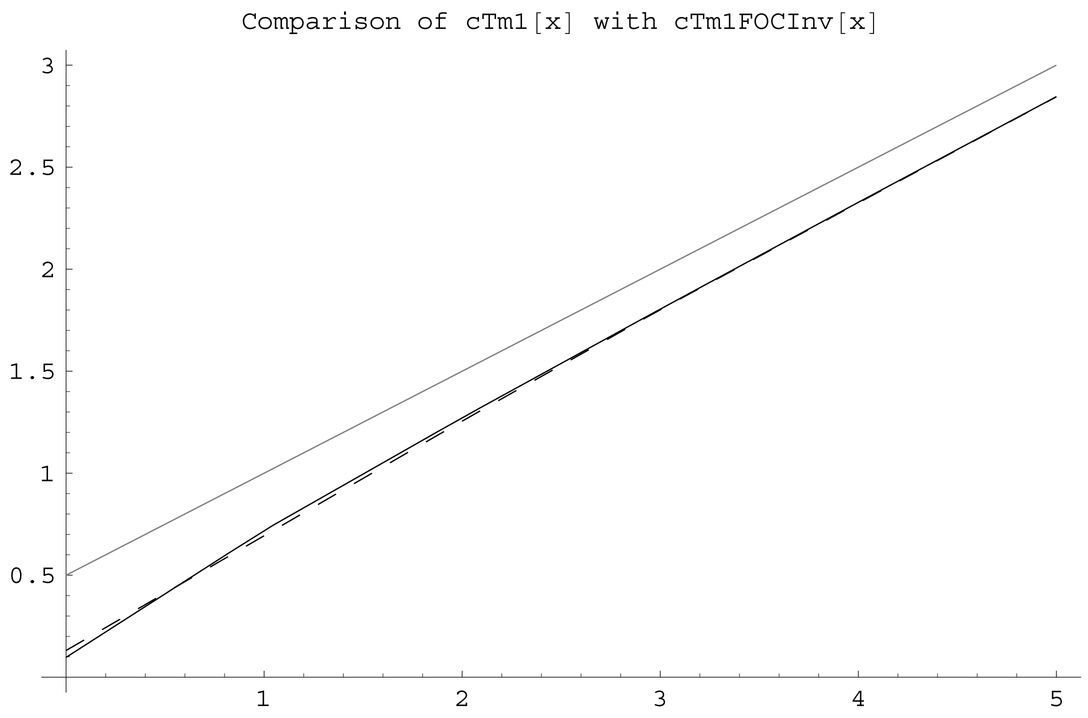

Figure 11:

5.6 Constraints

Problems of this type often come with additional constraints that must be satisfied. The most common type of constraint is a liquidity constraint that prevents the consumer's net worth from falling below some value, often zero.

With the additional constraint, the problem can be rewritten

$$V_{T-1}(x_{T-1}) = \max_{\{c_{T-1}\}} u(c_{T-1}) + \beta E_{T-1} V_T(x_T) \tag{60}$$

$$\text{s.t.} \quad s_{T-1} = x_{T-1} - c_{T-1} \tag{61}$$

$$x_T = R s_{T-1} + Y_T \tag{62}$$

$$s_{T-1} \geq 0 \tag{63}$$

By definition, the constraint will bind in those circumstances where the unconstrained consumer would behave in such a way as to violate the constraint. In this case, that means that the constraint binds if the level of consumption that satisfies the FOC:

$$c_{T-1}^{-\rho} = \beta \Omega'_{T-1}(x_{T-1} - c_{T-1}) \tag{64}$$

is greater than x_{T-1}. Call the value of c_{T-1} that satisfies this equation $\check{c}_{T-1}$. Then the constrained optimal level of consumption will be

$$c_{T-1}(x_{T-1}) = \min[x_{T-1}, \check{c}_{T-1}(x_{T-1})] \tag{65}$$

The introduction of the constraint also introduces a sharp nonlinearity in all of the functions at the point where the constraint begins to bind. As a result, to get solutions that are anywhere close to numerically accurate it is useful to augment the grid of values of the state variable to include the exact value at which the constraint becomes binding. Fortunately, the value of this point is relatively easy to calculate. We know that when the constraint is binding the consumer is saving nothing. We know the marginal value of saving at the point of zero saving is given by $\Omega'_{T-1}(0)$. Finally, we know that when the constraint is binding, $c_{T-1} = x_{T-1}$. Thus, the largest value of consumption for which the constraint is binding will be the point for which the marginal utility of consumption is exactly equal to the (expected, discounted) marginal value of saving. We know this because the marginal utility of consumption is a downward-sloping function and so if the consumer were to consume ϵ more, the marginal utility of that extra consumption would be *below* the (discounted, expected) marginal utility of saving, and thus the consumer would engage in positive saving and the constraint would no longer be binding. Thus the level of x_{T-1} at which the constraint begins to bind is:

$$\begin{aligned} u'(x_{T-1}) &= \beta \Omega'_{T-1}(0) \\ x_{T-1} &= (\beta \Omega'_{T-1}(0))^{(-1/\rho)}. \\ x_{T-1} &= \beta^{-1/\rho} \mho'_{T-1}(0). \end{aligned} \tag{66}$$

Note that once we have constructing the interpolating function $\hat{\mho}_{T-1}$ this expression is fast and easy to calculate. Once the value that solves this equation is calculated, we simply add that value of X to the vector xGrid and treat the new point just like any other point in xGrid.

The program that solves the constrained problem is `2period_intexp_foc_inv_constr.m.`; the resulting consumption rule is shown in figure 12 For comparison purposes, the approximate consumption rule from figure 11 is reproduced here as the dashing line. As expected, the liquidity constraint only causes a divergence between the two functions at the point where the optimal unconstrained consumption rule runs into the 45 degree line.

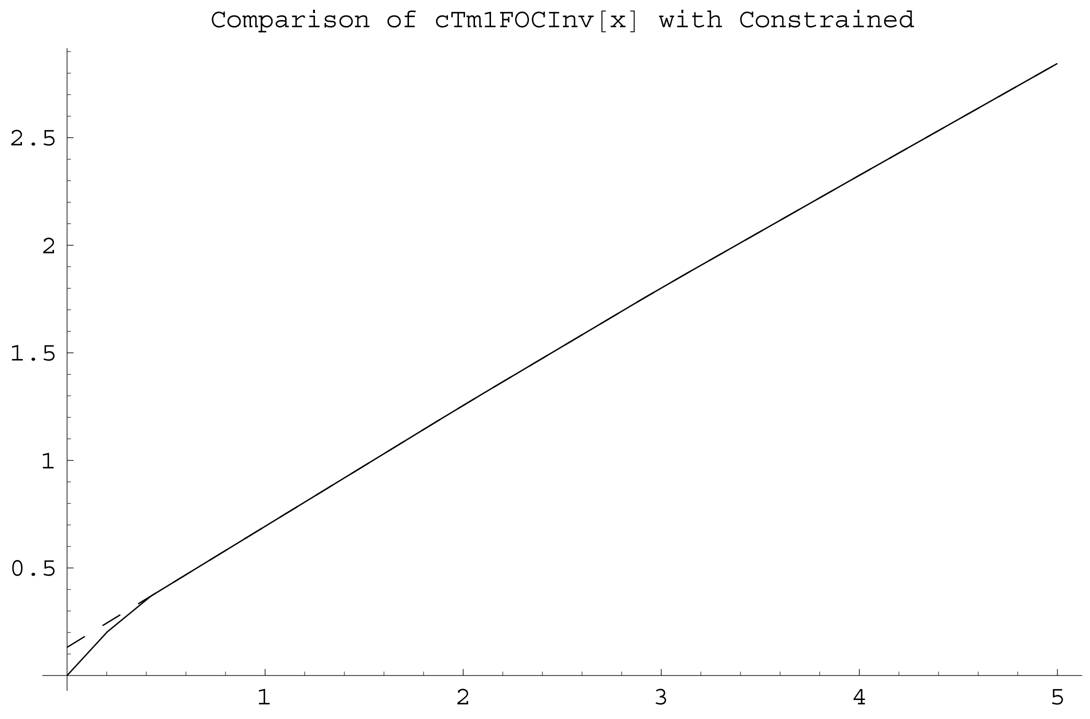

Figure 12:

6 Recursion

6.1 Theory

We have now learned how to construct an approximation to the value function $V_{T-1}(x_{T-1})$ and optimal consumption function $c_{T-1}(x_{T-1})$. How do we proceed back to earlier periods of life?

Recall equations (40) and (41):

$$\begin{aligned} \Omega_t^{'}(s_t) &= E_t[Ru'(c_{t+1}[Rs_t + \tilde{\epsilon}_{t+1}])] && (67) \\ u'(c_t) &= \beta\Omega_t^{'}(x_t - c_t). && (68) \end{aligned}$$

Assuming the problem has been solved up to period $t+1$ (and thus assuming that we have a numerical function $\hat{c}_{t+1}(x_{t+1})$), the first of these tells us how to calculate $\Omega_t(s_t)$, and the second tells us how to calculate $\hat{c}_t(x_t)$ given $\Omega_t(s_t)$. Our solution method essentially involves using these two equations in succession to work back progressively from period $T-1$ to the beginning of life. Stated generally, the method is as follows.

1. For the grid of values $s_{i,t}$ in sGrid, numerically calculate the value of $\mho_t^{'}(s_{i,t})$,

$$\begin{aligned} \mho_t^{'}(s_{i,t}) &= \left(\Omega_t^{'}(s_{i,t})\right)^{-1/\rho}, && (69) \\ &= \left(E_t\left[R(\hat{c}_{t+1}(Rs_{i,t} + \tilde{\epsilon}_{t+1}))^{-\rho}\right]\right)^{-1/\rho}, && (70) \end{aligned}$$

   generating a list of values $\mho_{i,t}^{'}$.

2. Construct an interpolating function $\hat{\mho}_t^{'}(s_t)$ that 'connects the dots' of $\{\{s_{1,t}, \mho_{1,t}^{'}\}, \{s_{2,t}, \mho_{2,t}^{'}\}, \ldots\}$.

3. Calculate the value of x_t at which the liquidity constraint begins to bind from the analogue to equation (66):

$$x_t \quad = \quad \beta^{-1/\rho}\mho'_t(0). \tag{71}$$

   and augment xGrid by adding this point to it.

4. Use a numerical rootfinding routine to solve the equation

$$c_t \quad = \quad \beta^{-1/\rho}\hat{\mho}'_t(x_t - c_t) \tag{72}$$

   at each of the values of xGrid, generating a list of values $\{\{x_{1,t}, c_{1,t}\}, \{x_{2,t}, c_{2,t}\}, \ldots\}$.

5. Interpolate between these solved points to obtain $\hat{c}_t(x_t)$.

Once we have $\hat{c}_t(x_t)$ we can continue the backwards recursion to period $t-1$ and so on back to the beginning of life.

Note that this loop does not contain steps for constructing $\hat{\Omega}'_t(s_t)$, $\hat{V}'_t(x_t)$, or $\hat{\Lambda}_t(x_t)$. This is because with $\hat{\mho}_t(s_t)$ and $\hat{c}_t(x_t)$ in hand, we simply *define* $\hat{\Omega}'_t(s_t) = [\hat{\mho}'_t(s_t)]^{-\rho}$, $\hat{V}'_t(x_t) = u'(\hat{c}_t[x_t])$, and $\hat{\Lambda}'_t = [\hat{V}'(x_t)]^{-1/\rho} = [u'(\hat{c}_t(x_t))]^{-1/\rho} = \hat{c}_t(x_t)$ so there is no need to construct interpolating functions for these functions - they arise 'free' (or nearly so) from our calculations for $\hat{\mho}'_t(s_t)$ and $\hat{c}_t(x_t)$.

The program `multiperiod.m` presents a fairly general and flexible approach to solving problems of this kind. The essential structure of the program is a loop which simply performs the recursion described above from the last period of 'life' back to period 1, where the variable "LifeLength" is defined in `multiperiod_vars.m` determines how many periods there are.

6.2 Mathematica Background

Mathematica has several features that are useful in solving the multiperiod problem.

- It can treat a user-created function as an object just like a number or a letter
- Mathematica uses the 'list' as its basic data structure. A Mathematica 'list' is a very powerful and flexible data construct. A list of length N in Mathematica can hold essentially anything in each of its N positions - a function, a number, another list, a symbolic expression, or any other object that Mathematica can recognize. The items at position i in a list named ExampleList are retrieved or addressed using the syntax `ExampleList[[i]]`.
- The function `Apply[FuncName_, DataListName_]` takes the function whose name is FuncName (for example, Vt) and the data in DataList (for example, $\{1, 19\}$) and returns the result that would have been returned by calling the function Vt[1,19].
- The function `Map[FuncToApply_,DataToApplyItTo_]` takes a list of possible arguments to the function FuncToApply and applies that function to each of the elements of that list sequentially. For example, `Map[Sin,{1,2,3}]` would return a list `{Sin[1],Sin[2],Sin[3]}`.

6.3 Program Structure

After the usual initializations, the heart of the program works like this.

6.3.1 Setup

First, in `multiperiod_functions.m`, the various "Raw" functions are defined. As above, a "Raw" function is the exact representation of one of the theoretical constructs defined in the theory above. For example, `OmegaInvPrimetRaw[x_,LifePosn_]` (recall that 'Inv' is shorthand for the inverted version of the function) is defined as

$$\begin{aligned} \mho_t'(s_t) &= \left(E_t\left[R(\hat{c}_{t+1}(Rs_t+\tilde{\epsilon}_{t+1}))^{-\rho}\right]\right)^{-1/\rho} && (73)\\ &= \left(\frac{1}{n}\sum_{i=1}^{n} R\left[(\hat{c}_{t+1}(Rs_t+\epsilon_i))^{-\rho}\right]\right)^{-1/\rho}. && (74)\end{aligned}$$

Second, in `multiperiod_vars.m`, a variety of data structures that will be used in the iteration are set up. For example, for `OmegaInvPrimet`, the following variables are defined.

- The function `MakeArgArrays` creates `OmegaInvPrimetArgArray`, a list that contains all the possible combinations and permutations of the arguments to `OmegaInvPrimet`. Since `OmegaInvPrimet` has only one argument, the level of cash-on-hand x_t (ignoring the 'LifePosn' argument), in this case `OmegaInvPrimetArgArray` boils down to the list of possible values of xGrid. But if there were two state variables, x and h, say, then `OmegaInvPrimetArgArray` would contain $\{\{x_1,h_1\},\{x_1,h_2\}\ldots,\{x_2,h_1\},\{x_2,h_2\}\ldots\}$ and so on. Similar data constructs are created for all the other functions `(VInvPrimet, OmegaInvt, OmegaInvPrimet, ct)` the program will be creating.
- The functions `MakeNewVar` ... create a set of 'lists' of length `LifeLength`. The most important of these are the InterpData and InterpFunc objects. Again using `OmegaInvPrimet` as an example, the program creates
  - `OmegaInvPrimetInterpData[[LifePosn]]` will hold a list containing the value of `OmegaInvPrimet` at each point defined in `OmegaInvPrimetArgArray`
  - `OmegaInvPrimetInterpFunc[[LifePosn]]` will hold the `InterpolatingFunction` object which is created from the data in `OmegaInvPrimetInterpData`

6.3.2 Iteration

After setting up an indicator variable LifePosn which will always represent the period of life that the program is currently attempting to solve, and preserving a baseline set of values for xGrid (for reasons that will become apparent momentarily), the program sets up a "While" loop that counts down from the last period of life. The structure of the "While" loop is as follows

1. Construct the `InterpolatingFunction` for `OmegaInvt` for period LifePosn
   - In `OmegaInvtArgumentList` create a list with the structure $\{\{x_1,\text{LifePosn}\},\ldots,\{x_n,\text{LifePosn}\}\}$
   - In `OmegaInvtResultsRaw` collect the results of applying the function `OmegaInvtRaw` to the arguments in `OmegaInvtArgumentList`,
     `OmegaInvtResultsRaw` $=\{$`OmegaInvtRaw[`x_1`,LifePosn]`,`OmegaInvtRaw[`x_2`,LifePosn]`$,\ldots\}$
   - Construct and put into the waiting LifePosn'th slot of the variable `OmegaInvtInterpData` a list of the form $\{\{x_1,\texttt{OmegaInvtRawResults}_1\},\{x_2,\texttt{OmegaInvtRawResults}_2\},\ldots\}$
   - From the data in `OmegatInterpData[[LifePosn]]` construct an `InterpolatingFunction` object and put it in `OmegatInterpFunc[[LifePosn]]`

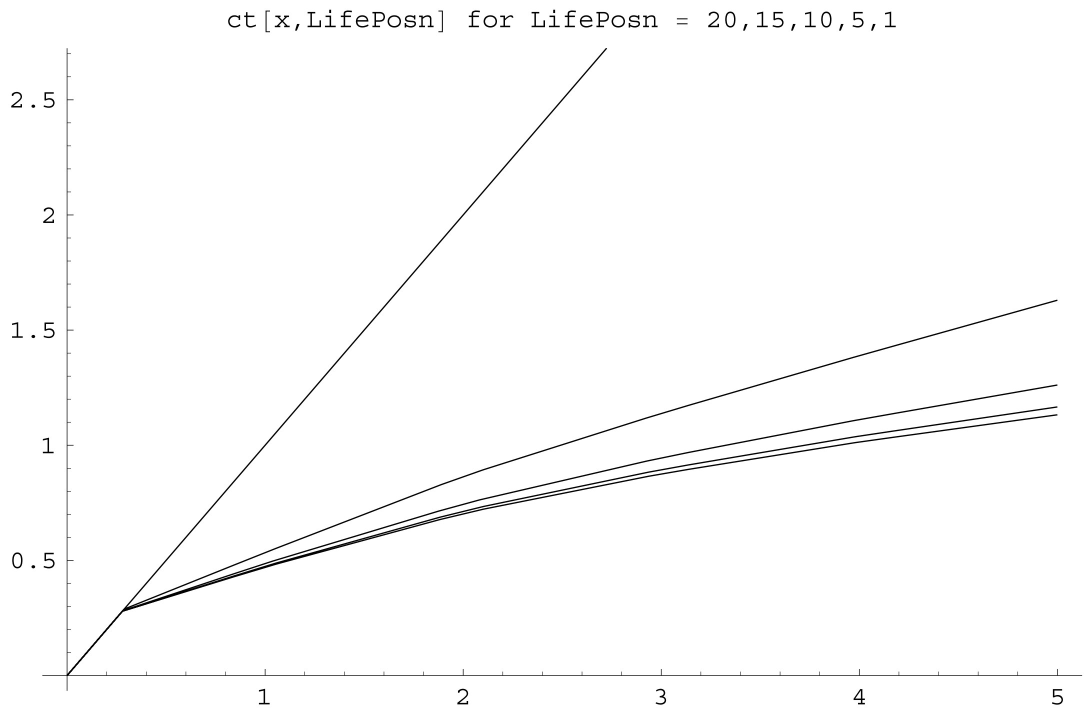

Figure 13:

2. Construct the `InterpolatingFunction` for `OmegaInvPrimet` for period LifePosn

   - Follow all of the same steps as for `OmegaInvt`

3. Calculate the point at which the liquidity constraint begins to bind using equation (71) and add this point to the list of values in xGrid.

4. Recreate the ArgArray arrays for functions that take x as an argument to reflect the inclusion of the new point

5. Construct `ctArgumentList` as above, then map the `ctArgumentList` into the function ctRaw to find the values of $c_{i,t}$ which satisfy equation (72). From the list of $\{x_{i,t}, c_{i,t}\}$ pairs construct the `InterpolatingFunction` object that represents $\hat{c}_t(x_t)$.

6. Decrement `LifePosn` and repeat loop if `LifePosn` > 0.

6.4 Results

As written, the program creates three `InterpolatingFunctions`, for $\hat{\mho}_t(s_t), \hat{\mho}'_t(s_t)$, and $\hat{c}_t(x_t)$, and all other functions are defined in terms of these functions (although $\hat{\mho}_t(s_t)$ is not actually used in the solution of the problem, since it is solved using the first order conditions. These functions can be evaluated in any period for any value of x. For values of x outside of the grid encompassed in xGrid and sGrid, the program extrapolates from the relationship between the nearest two points in the grids.

As an illustration, figure 13 shows $\hat{c}_t(x_t)$ for $t = \{20, 15, 10, 5, 1\}$. At least one feature of this figure is encouraging: the consumption functions converge as one recedes from the end of life, something that Carroll (1996) shows must be true under certain parametric conditions that are satisfied by this problem.

7 Multiple State Variables

We now wish to consider how the problem changes if there are multiple state variables rather than just a single state variable. The example we will use will be the case where the utility from consumption depends on the size of a 'habit stock' which represents an average of past levels of consumption. Formally, the goal is to

$$\max_{\{c_t\}} \quad \sum_{s=t}^{T} \beta^{s-t} u(c_s, h_{s-1}) \tag{75}$$

Now there are two state variables in the problem at time t, the level of assets x_t and the level of the habit stock h_{t-1}, where the accumulation equation for x_t is the same as before and the transition equation for habits is

$$h_t \quad = \quad h_{t-1} + \lambda(c_t - h_{t-1}). \tag{76}$$

That is, the habit stock at the end of this period is equal to the habit stock at the end of last period plus a proportion of the gap between the level of consumption chosen this period and the level of the habit stock from last period. In other words, habits 'catch up' to consumption at rate λ.

Assume that the utility function is given by

$$u(c_t, h_{t-1}) = \frac{(c_t/h_{t-1}^{\gamma})^{1-\rho}}{1-\rho} \tag{77}$$

Now that u_t has two arguments we need to be able to distinguish between the derivatives with respect to each argument. Our notation will be that the derivative of u_t with respect to c_t is $u_t^c(c_t, h_{t-1})$ or u_t^c for short, and analogously for u_t^h. Thus we have

$$\begin{aligned}
u_t^c &= (c_t h_{t-1}^{-\gamma})^{-\rho} h_{t-1}^{-\gamma} && (78)\\
&= c_t^{-\rho} h_{t-1}^{\rho\gamma-\gamma} && (79)\\
u_t^h &= -\gamma(c_t h_{t-1}^{-\gamma})^{-\rho} c_t h_{t-1}^{-\gamma-1} && (80)\\
&= -\gamma c_t^{1-\rho} h_{t-1}^{\gamma\rho-\gamma-1} && (81)
\end{aligned}$$

Bellman's equation for this problem (imposing liquidity constraints again) is

$$\begin{aligned}
&V_t(x_t, h_{t-1}) = \\
&\max_{\{c_t\}} \quad u(c_t, h_{t-1}) \quad +\beta E_t[V_{t+1}(x_{t+1}, h_t)] && (82)\\
&\qquad \text{such that} \\
x_{t+1} &= R[x_t - c_t] + \epsilon_{t+1} && (83)\\
h_t &= h_{t-1} + \lambda(c_t - h_{t-1}) && (84)\\
c_t &\leq x_t && (85)
\end{aligned}$$

As was done for utility above, define the derivatives of V_t with respect to each argument as $V_t^x(x_t, h_{t-1})$ or V_t^x for short, and analogously for V_t^h. Also as above, we want to define a function which corresponds to the expectation of the value of doing ending period t in a given position, but now the 'position' involves both the level of savings s_t and the level of the habit stock h_t:

$$\Omega_t(s_t, h_t) \quad = \quad E_t[V_{t+1}(Rs_t + \tilde{\epsilon}_{t+1}, h_t))] \tag{86}$$

For future reference note that the derivatives are

$$\Omega_t^s = E_t[RV_{t+1}^x] \tag{87}$$

$$\Omega_t^h = E_t[V_{t+1}^h] \tag{88}$$

and the maximization problem can be rewritten

$$\begin{aligned} V_t(x_t, h_{t-1}) = & \\ \max_{\{c_t\}} \quad u(c_t, h_{t-1}) & \quad + \beta\Omega_t(x_t - c_t, h_{t-1} + \lambda(c_t - h_{t-1})) \\ & \text{such that} \\ c_t \quad & \leq \quad x_t \end{aligned}$$

7.1 Optimality Conditions

7.1.1 The First Order Condition for c_t

The FOC for this problem with respect to c_t is:

$$0 = u_t^c + \beta E_t\left[V_{t+1}^x(-R) + V_{t+1}^h\lambda\right] \tag{89}$$

$$u_t^c = \beta E_t[RV_{t+1}^x - \lambda V_{t+1}^h] \tag{90}$$

$$= \beta\left[\Omega_t^s - \lambda\Omega_t^h\right] \tag{91}$$

Substituting the definition of u_t^c:

$$c_t^{-\rho}h_{t-1}^{\rho\gamma-\gamma} = \beta\left[\Omega_t^s - \lambda\Omega_t^h\right] \tag{92}$$

$$c_t = \left[h_{t-1}^{\gamma-\rho\gamma}\beta\left(\Omega_t^s - \lambda\Omega_t^h\right)\right]^{-1/\rho} \tag{93}$$

and the liquidity constraint implies that if the $\check{c}_t$ which satisfies this equation is larger than x_t the consumer spends x_t rather than $\check{c}_t$. The point at which the liquidity constraint becomes binding is implicitly defined by the equation

$$x_t = \left[h_{t-1}^{\gamma-\rho\gamma}\beta\left(\Omega_t^s(0, h_{t-1} - \lambda(x_t - h_{t-1})) - \lambda\Omega_t^h(0, h_{t-1} - \lambda(x_t - h_{t-1})\right)\right]^{-1/\rho}, \tag{94}$$

which must be solved numerically (in contrast to the situation in the problem without habits). Note that this equation implies that the liquidity constraint becomes binding at a different value of x_t for every possible different value of h_{t-1}.

7.1.2 Applying the Envelope Theorem

The Envelope theorem on x_t says:

$$V_t^x = \frac{\partial V_t}{\partial x_t} + \overbrace{\frac{\partial V_t}{\partial c_t}}^{=0}\frac{\partial c_t}{\partial x_t} \tag{95}$$

$$V_t^x = \beta E_t[RV_{t+1}^x] \tag{96}$$

Substituting this into the FOC equation (90) gives

$$u_t^c = V_t^x - \beta E_t[\lambda V_{t+1}^h] \tag{97}$$

$$V_t^x = u_t^c + \beta E_t[\lambda V_{t+1}^h] \tag{98}$$

$$= u_t^c + \beta\lambda\Omega_t^h \tag{99}$$

What if the consumer is liquidity constrained? It is useful here to rewrite Bellman's equation:

$$V_t(x_t, h_{t-1}) \quad = \quad u(c_t, h_{t-1}) + \beta E_t \left[V_{t+1}(R[x_t - c_t] + \tilde{\epsilon}_{t+1}, h_{t-1} + \lambda(c_t - h_{t-1}))\right]$$

Substituting in the fact that $c_t = x_t$ (because the consumer is constrained)

$$V_t(x_t, h_{t-1}) \quad = \quad u(c_t, h_{t-1}) + \beta E_t \left[V_{t+1}(\tilde{\epsilon}_{t+1}, h_{t-1} + \lambda(c_t - h_{t-1}))\right]$$

Thus $\partial V_t / \partial x_t = 0$, and because the liquidity constraint implies that $\partial c_t / \partial x_t = 1$, equation (95) becomes

$$\begin{aligned} V_t^x &= \frac{\partial V_t}{\partial c_t} && (100) \\ &= u_t^c + \beta E_t[\lambda V_{t+1}^h] && (101) \end{aligned}$$

which is identical to the expression (98) for V_t^x for the unconstrained consumer.

The Envelope theorem for h_{t-1} says:

$$\begin{aligned} V_t^h &= \frac{\partial V_t}{\partial h_{t-1}} + \overbrace{\frac{\partial V_t}{\partial c_t}}^{=0} \frac{\partial c_t}{\partial h_{t-1}} \\ &= u_t^h + \beta E_t \left[V_{t+1}^h \frac{\partial h_t}{\partial h_{t-1}} \right] \\ &= u_t^h + \beta E_t[(1-\lambda) V_{t+1}^h] \\ &= u_t^h + \beta \Omega_t^h && (102) \end{aligned}$$

What if the consumer is constrained? In that case while $\partial V_t / \partial c_t \neq 0$, $\partial c_t / \partial h_{t-1} = 0$, so as with V_t^x the constraint has no effect on the expression for V_t^h.

7.2 Transformations

Note one inconvenient feature of these equations: there is no longer a uniquely appropriate 'transformation' for the problem, as there was in the first problem. To see this, consider the last period of life and suppose there is no uncertainty. Then $V_T^x = u_T^c$ will be of the form $c_T^{-\rho} h_{T-1}^{\rho\gamma - \gamma}$, while $V_T^h = u^h$ will be something of the form $-\gamma c_T^{1-\rho} h_{T-1}^{\gamma\rho - \gamma - 1}$. It is possible to exponentiate either of these equations to make it linear in one or the other of c or h, but not both. In practice, it is best to transform both equations so that they are linear in c, because the habit stock h will 'catch up' to c and thus is not likely to stray very far from c anyhow. Thus the transformations I use are[3]

$$\begin{aligned} \mho_t^s(s_t, h_t) &= [\Omega_t^s(s_t, h_t)]^{-1/\rho} && (103) \\ \Lambda_t^x(x_t, h_{t-1}) &= [V_t^x(x_t, h_{t-1})]^{-1/\rho} && (104) \\ \mho_t^h(s_t, h_t) &= [-\Omega_t^h(s_t, h_t)/\gamma]^{1/(1-\rho)} && (105) \\ \Lambda_t^h(x_t, h_{t-1}) &= [-V_t^h(x_t, h_{t-1})/\gamma]^{1/(1-\rho)} && (106) \\ \Lambda_t(x_t, h_{t-1}) &= [(1-\rho) V_t(x_t, h_{t-1})]^{1/(1-\rho)} && (107) \end{aligned}$$

7.3 The Program

The consumption problem with habit formation is solved in `habits.m`, whose structure closely follows that of `multiperiod.m`.

[3]If it were necessary, it would be possible to do more sophisticated transformations, but in practice these work reasonably well.

7.3.1 Setup

Assuming the problem has been solved up to period $t+1$ (and thus we have numerical functions $\hat{V}^x_{t+1}(x_{t+1}, h_t)$ and $\hat{V}^h_{t+1}(x_{t+1}, h_t)$,

1. Form a list called `ArgArray` of all possible combinations of the values in sGrid and hGrid, and index the components of that list by i. Thus if there are m points in both grids we have `ArgArray`=

   $\{\{s_1, h_1\}, \{s_1, h_2\}, \ldots, \{s_1, h_m\}, \{s_2, h_1\}, \{s_2, h_2\}, \ldots, \{s_2, h_m\}, \{s_m, h_1\}, \{s_m, h_2\}, \ldots, \{s_m, h_m\}\}$.

   At each of the $\{s, h\}$ combinations in `ArgArray` calculate the value of $\mho^s_t$ and $\mho^h_t$ (from equations (87) and (88)),

$$\mho^s_t(s_{i,t}, h_{i,t}) = \left(E_t\left[R(\hat{V}^x_{t+1}(Rs_{i,t} + \tilde{\epsilon}_{t+1}, h_{i,t}))\right]\right)^{-1/\rho}, \tag{108}$$

$$\mho^h_t(s_{i,t}, h_{i,t}) = \left(-E_t\left[\hat{V}^h_{t+1}(Rs_{i,t} + \tilde{\epsilon}_{t+1}, h_{i,t})\right]/\gamma\right)^{1/(1-\rho)}, \tag{109}$$

   generating lists of values $\mho^s_{i,t}$,$\mho^h_{i,t}$.

2. Construct interpolating functions $\hat{\mho}^s_t(s_t, h_t)$ and $\hat{\mho}^s_t(s_t, h_t)$ by connecting the dots, from which we can obtain $\hat{\Omega}^s$ and $\hat{\Omega}^h$ via the inverse of the transformations (103) and (105).

3. Use equation (94) to find the point at which the liquidity constraint becomes binding for the largest and smallest values of h in hGrid, and augment xGrid with those values of x.

4. Use a numerical rootfinding routine to find the c_t that solves equation (93):

$$c_t = \left[h^{\gamma-\rho\gamma}_{t-1}\beta\left(\hat{\Omega}^s_t(x_t - c_t, h_{t-1} + \lambda(c_t - h_{t-1})) - \lambda\hat{\Omega}^h_t(x_t - c_t, h_{t-1} + \lambda(c_t - h_{t-1}))\right)\right]^{-1/\rho}$$

   at each of the values of xGrid, generating a list of values $\{\{x_{1,t}, c_{1,t}\}, \{x_{2,t}, c_{2,t}\}, \ldots\}$.

5. Interpolate between these solved points to obtain $\hat{c}_t(x_t)$.

6. Use equations (99) and (102) and the transformations (104) and (106) to construct interpolating approximations to the inverse marginal value functions:

$$\Lambda^x_t = \left[u^c_t + \beta\lambda\hat{\Omega}^h_t\right]^{-1/\rho} \tag{110}$$

$$\Lambda^h_t = \left(-\left[u^h_t + \beta\hat{\Omega}^h_t\right]/\gamma\right)^{1/(1-\rho)} \tag{111}$$

   from which we can construct $\hat{\Lambda}^x_t$ and $\hat{\Lambda}^h_t$ via the usual Map-then-interpolate strategy. $\hat{V}^x_t$ and $\hat{V}^h_t$ are then defined via the inverse of $\hat{V}^x_t$ and $\hat{V}^h_t$ via the transformations (104) and (106).

Thus we have generated $\hat{V}^s_t$ and $\hat{V}^h_t$ from $\hat{V}^s_{t+1}$ and $\hat{V}^h_{t+1}$, and we can continue the iteration until we reach the first period of life.

Note that the first five of these steps are simply the generalization of the five steps used in solving the single-state variable problem outlined earlier. Recall that in solving the single-state-variable problem we argued that it was not necessary to construct an interpolating function for $\hat{\Lambda}'_t$ because we could obtain the value of $\hat{V}'_t$ directly from the $\hat{c}_t$ function via the relation $\hat{V}'_t(x_t) = u'[\hat{c}_t(x_t)]$. The analogous equation here is $\hat{V}^x_t(x_t, h_{t-1}) = u^c(\hat{c}_t(x_t, h_{t-1}), h_{t-1}) + \beta\lambda\hat{\Omega}^h(x_t - c_t, h_{t-1} + \lambda(c_t - h_{t-1}))$. The problem with obtaining $\hat{V}^x_t$ from this equation is that Ω^h_t is a negative number. As a result, if the extrapolation error for Ω^h_t happens to exceed that for $u'[\hat{c}_t]$ it is possible that when $u'[\hat{c}_t] + \hat{\Omega}^h_t$ is evaluated at a point outside of the grid, the sum could be negative - which is economic nonsense: giving an agent more wealth can never actually *reduce*

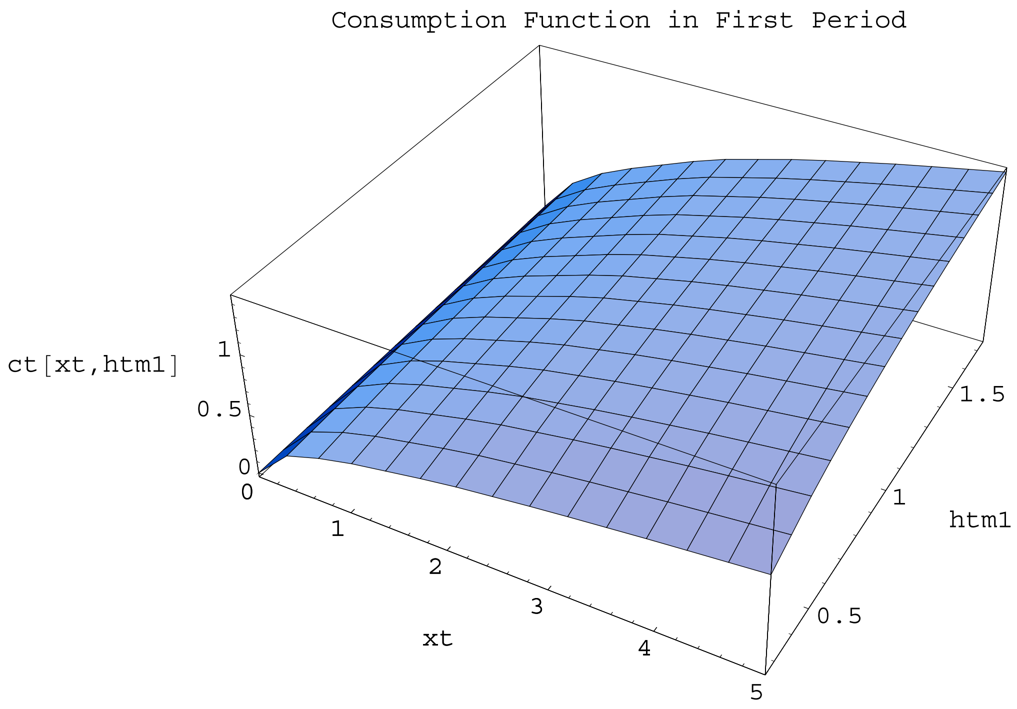

Figure 14:

utility, because the agent could just give it away. Furthermore, a negative value of $\hat{V}_t^x$ can wreak havoc on the program - for example, when its inverse is taken. This problem does not arise if $\hat{\Lambda}_t^s$ is constructed for the gridpoints and the value of $\hat{V}_t^s$ obtained from inverting the extrapolated value of $\hat{\Lambda}_t^s$.

The problem is solved in the program `habits.m`. Details of the Mathematica implementation follow those described above for `multiperiod.m` very closely, and so need not be detailed here. The program generates a three-D figure showing the consumption rule $c_t(x_t, h_{t-1})$ for the first period of 'life.' The figure behaves as one would expect: consumption is increasing in the level of resources and in the level of the habit stock.

8 Multiple Control Variables

We now consider how to solve problems with multiple control variables. To keep matters as simple as possible, we will revert to the case where there is a single state variable; the combination of multiple states and multiple controls is a straightforward combination of the techniques of this section and the previous one.

8.1 Theory

The new control variable that we will assume the consumer can choose is the portion of the portfolio to invest in risky versus safe assets. Designating the gross return on the risky asset between period t and $t+1$ as $R_{e,t+1}$ (where the 'e' is meant as a mnemonic for 'equities,' the risky asset usually considered in models of this type), and using $w_{e,t}$ to represent the proportion of the portfolio (w is mnemonic for 'weight') invested in equities between t and $t+1$, and continuing to use R for the rate of return on the riskless asset, the overall return on the consumer's portfolio between t and $t+1$ will be

$$\begin{aligned} R_{t+1} &= R(1 - w_{e,t}) + R_{e,t+1} w_{e,t} && (112) \\ &= R + (R_{e,t+1} - R) w_{e,t} && (113) \end{aligned}$$

and the maximization problem is

$$\begin{aligned} V_t(x_t) = & \\ \max_{\{c_t, w_{e,t}\}} \quad u(c_t) + & \quad \beta E_t[V_{t+1}(x_{t+1})] \quad (114) \\ & \text{such that} \\ x_{t+1} \quad = & \quad R_{t+1}[x_t - c_t] + \epsilon_{t+1} \quad (115) \\ R_{t+1} \quad = & \quad R + (R_{e,t+1} - R)w_{e,t} \quad (116) \\ 0 \leq \quad w_{e,t} & \quad \leq 1 \quad (117) \end{aligned}$$

or

$$\begin{aligned} V_t(x_t) = & \\ \max_{\{c_t, w_{e,t}\}} \quad u(c_t) + & \quad \beta E_t[V_{t+1}(\tilde{R}_{t+1}[x_t - c_t] + \tilde{\epsilon}_{t+1})] \quad (118) \\ & \text{such that} \quad (119) \\ 0 \leq \quad w_{e,t} & \quad \leq 1 \quad (120) \end{aligned}$$

The first order condition with respect to c_t is almost identical to that in the single-control problem, equation (31), with the only difference being that the nonstochastic interest rate R is now replaced by $\tilde{R}_{t+1}$:

$$u'(c_t) \quad = \quad \beta E_t[\tilde{R}_{t+1} V'_{t+1}(x_{t+1})] \quad (121)$$

and the Envelope theorem derivation remains the same so that we still have

$$u'(c_t) \quad = \quad V_t^{'}(x_t) \quad (122)$$

implying the Euler equation for consumption

$$u'(c_t) \quad = \quad \beta E_t[\tilde{R}_{t+1} u'(c_{t+1})]. \quad (123)$$

The first order condition with respect to the risky portfolio share is

$$\begin{aligned} 0 \quad &= \quad E_t[V_{t+1}^{'}(x_{t+1})(\tilde{R}_{e,t+1} - R)] \quad (124) \\ &= \quad E_t[u^{'}(c_{t+1}[x_{t+1}])(\tilde{R}_{e,t+1} - R)] \quad (125) \end{aligned}$$

As before, it will be useful to define Ω_t as a function which yields the expected value as of ending period t in a given state. However, now that there are two control variables, the expectation must be defined as a function of the choices of both of those variables, because the expectation as of time t of value as of time $t+1$ will depend now not just on how much the agent saves, but also on how those savings are allocated between the risky and riskless assets. Thus we define

$$\Omega_t(s_t, w_{e,t}) \quad = \quad E_t[V_{t+1}(x_{t+1})]$$

which has derivatives

$$\begin{aligned} \Omega_t^s \quad &= \quad E_t[\tilde{R}_{e,t+1} V_{t+1}^x(x_{t+1})] \\ \Omega_t^w \quad &= \quad E_t[(\tilde{R}_{e,t+1} - R) V_{t+1}^x(x_{t+1})] \end{aligned}$$

implying that the first order conditions (123) and (125) and can be rewritten

$$\begin{aligned} u'(c_t) \quad &= \quad \beta \Omega_t^s(x_t - c_t), \quad (126) \\ 0 \quad &= \quad \Omega_t^w(s_t). \quad (127) \end{aligned}$$

8.2 Application

Our first step is to specify the stochastic process for $R_{e,t+1}$. We follow the common practice of assuming that returns are lognormally distributed, $\log R_e \sim \mathcal{N}(\mu - r, \sigma_e^2)$ where μ is the equity premium over the returns r available on the riskless asset.

As with labor income uncertainty, it is necessary to discretize the rate-of-return risk in order to have a problem that is soluble in a reasonable amount of time. We follow the same procedure as for labor uncertainty, generating a set of m equiprobable values of R_e which we will index by j, $R_{e,j,t+1}$.

Now let's rewrite the expressions for the derivatives of Ω_t explicitly:

$$\Omega_t^s(s_t, w_{e,t}) = \left(\frac{1}{mn}\right)\sum_{i=1}^{n}\sum_{j=1}^{m}\left[R_{e,j,t+1}\left(c_{t+1}(R_{e,j,t+1}s_t + \epsilon_i)\right)^{-\rho}\right] \tag{128}$$

$$\Omega_t^w(s_t, w_{e,t}) = \left(\frac{1}{mn}\right)\sum_{i=1}^{n}\sum_{j=1}^{m}\left[(R_{e,j,t+1} - R)\left(c_{t+1}(R_{e,j,t+1}s_t + \epsilon_i)\right)^{-\rho}\right]. \tag{129}$$

Writing these equations out explicitly makes a problem very apparent: for every different combination of $\{s_t, w_{e,t}\}$ that the routine wishes to consider, it must perform two double-summations of $m \times n$ terms. Once again, there is an inefficiency if it must perform these same calculations many times for the same or nearby values of $\{s_t, w_{e,t}\}$, and again the solution is to construct an approximation to the derivatives of the Ω function.

Details on the construction of the interpolating approximation are given below; assume for the moment that we have the approximations $\hat{\Omega}_t^s$ and $\hat{\Omega}_t^w$ in hand and want to proceed. As noted above, nonlinear equation solvers (including those built into Mathematica) can find the solution to a set of simultaneous equations. Thus we could ask Mathematica to solve

$$c_t^{-\rho} = \beta\hat{\Omega}_t^s(x_t - c_t, w_{e,t}) \tag{130}$$

$$0 = \hat{\Omega}_t^w(x_t - c_t, w_{e,t}) \tag{131}$$

simultaneously for the set of potential x_t defined in xGrid. However, multidimensional constrained maximization problems are difficult and sometimes quite slow to solve. There is a potentially better way. Define the problem

$$\Omega_{*,t}(s_t) = \max_{\{w_{e,t}\}} \Omega_t(s_t, w_{e,t}) \tag{132}$$

$$\text{such that}$$

$$0 \leq w_{e,t} \leq 1 \tag{133}$$

where the * subscript indicates that Ω has been optimized with respect to all of the arguments other than the one still present (s_t). We solve this problem for the set of gridpoints in sGrid and use the results to construct the interpolating function $\hat{\Omega}_{*,t}(s_t)$. With this function in hand, we can use the first order condition from the single-control problem

$$c_t^{-\rho} = \beta\hat{\Omega}_{*,t}^s(x_t - c_t).$$

to solve for the optimal level of consumption as a function of x_t. Thus we have transformed the multidimensional optimization problem into a sequence of two simple optimization problems for which solutions are much easier and more reliable.

Note the parallel between this trick and the fundamental insight of dynamic programming: dynamic programming techniques transform a multi-period (or infinite-period) optimization problem into a sequence of two-period optimization problems which are individually much easier to solve; we have done the same thing here, but with multiple dimensions of controls rather than multiple periods.

8.3 Implementation

The program which solves the problem with multiple control variables is `multicontrol.m`.

The first functions defined in `multicontrol_functions.m` correspond to $\Omega_t(s_t, w_{e,t})$ and its derivatives with respect to its arguments. Structurally these functions are very similar to the $\Omega_t(s_t, h_t)$ functions defined in `habits.m`; indeed, from the standpoint of the end of period t after the portfolio share has been chosen, the portfolio share is essentially a state variable, so the resemblance to the multistate problem is more than skin deep.

The first function definition that does not resemble anything in either `habits.m` or `multiperiod.m` is `OmegaInvOpttRaw[st_,LifePosn_]`. This function, for its input value of s_t, calculates the value of the portfolio share $w_{e,t}$ which satisfies the first order condition (131); tests whether the optimal portfolio share would violate the constraints, and if so resets the portfolio share to the constrained optimum. The function returns three results: the (inverse) value associated with the optimal choice of $w_{e,t}$ (designated $\mho_{*,i,t}$), the (inverse) marginal value of saving at the optimal $w_{e,t}$ ($\mho^s_{*,i,t}$), and the optimal value of the portfolio share itself, $w_{i,t}$, from which the functions $\hat{\mho}_{*,t}(s_t), \hat{\mho}^s_{*,t}(s_t)$ and $\hat{w}_t(s_t)$ will be constructed.

The subsequent function definitions in the file assume that these functions have been constructed and stored, respectively, in `OmegaInvOptt`,`OmegasInvOptt`, and `wt`, where the naming convention is obviously that 'Opt' stands for *. With $\hat{\Omega}^s_{*,t}(s_t)$ in hand the analysis is essentially identical to that for the standard multiperiod problem with a single control.

The structure of the program in detail is as follows. First, perform the usual initializations. Then initialize wGrid and the other variables specific to the multiple control problem.[4] In particular, there are now three kinds of functions: those with both s_t and $w_{e,t}$ as arguments, those with just s_t, and those with x_t.

Once the setup is complete, the heart of the program is the following loop.

1. Construct $\hat{\mho}_t(s_t, w_{e,t}), \hat{\mho}^s_t(s_t, w_{e,t})$, and $\hat{\mho}^w_t(s_t, w_{e,t})$ using the usual calculation and interpolation over the tensor defined by the combinations of the elements of sGrid and wGrid.

2. Call the function `OmegaInvOptt[st_,LifePosn_]` with the list of points in sGrid to generate $\mho_{*,i,t}, \mho^s_{*,i,t}$, and $w_{i,t}$, and generate the functions $\hat{\mho}_{*,t}, \hat{\mho}^s_{*,t}$, and $\hat{w}_t$ via interpolation.

3. Find the value of x_t where the liquidity constraint begins to bind, and augment xGrid with this point, then recreate the `ArgArrays` as usual.

4. Using $\hat{\Omega}_{*,t}(s_t)$ in place of $\hat{\Omega}_t(s_t)$, follow the same procedures as in `multiperiod.m` to generate $\hat{c}_t(x_t), \hat{\Lambda}_t(x_t)$, and $\hat{\Lambda}^x_t(x_t)$, from which $\hat{V}_t(x_t)$ and $\hat{V}^x_t(x_t)$ are derived, as usual, by inversion.

5. Decrement LifePosn; if LifePosn > 0, repeat the loop.

8.4 Results

Figure 15 plots the first-period 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 plots the optimal portfolio share as a function of the level of savings. This figure exhibits several interesting features. First, even with a coefficient of relative risk aversion of 10, an equity premium of only 4 percent, and an annual standard deviation in returns of 15 percent, the average level of the portfolio

[4]Note the choice of a coefficient of relative risk aversion of 10, in contrast with the choice of 2 made for the previous problems. This choice reflects the well-known 'stockholding puzzle,' which is the microeconomic equivalent of the equity premium puzzle: for plausible descriptions of income uncertainty, rate of return risk, and the equity premium, the typical consumer should hold all or nearly all of their portfolio in equities. Thus we choose an implausibly high value for the coefficient of relative risk aversion in order to generate portfolio structure behavior more interesting than a choice of 100 percent equities in every period for every level of wealth.

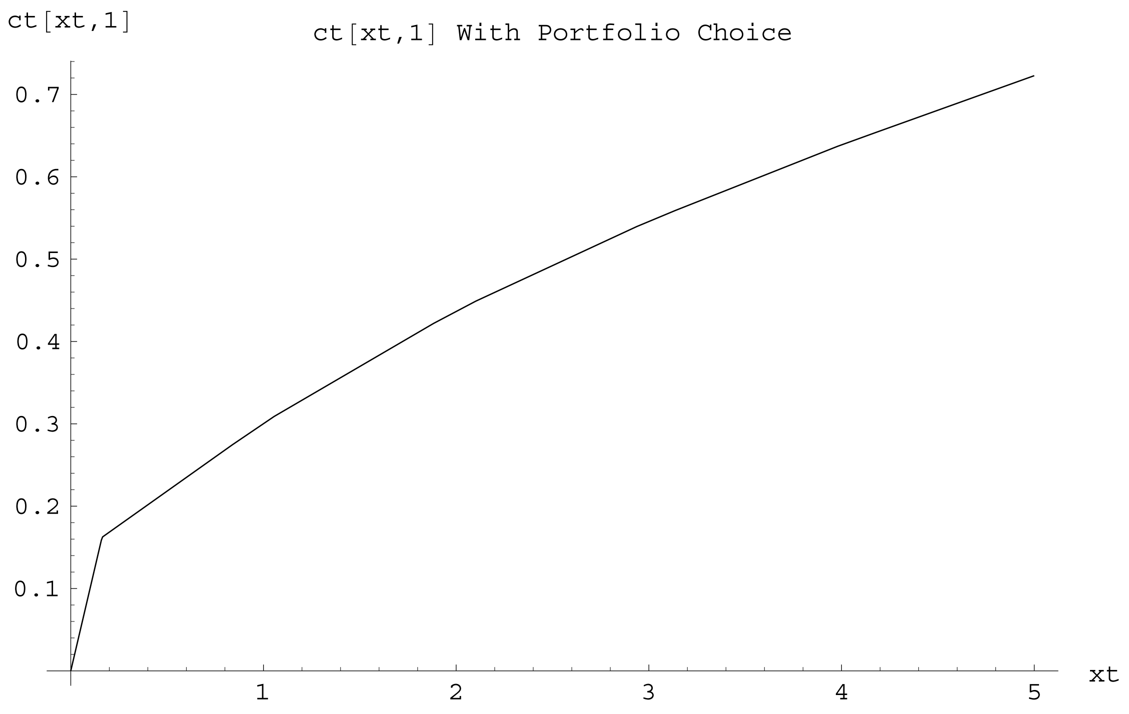

Figure 15:

share kept in stocks is almost 50 percent for most values of s_t. 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 resources in stocks, while the rich should be more cautious. This bizarre prediction is a consequence of the assumption about labor income risk. Those consumers who are poor in measured wealth are likely to 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, those with large amounts of current physical wealth will be financing a large proportion of future consumption out of that wealth, and hence 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.

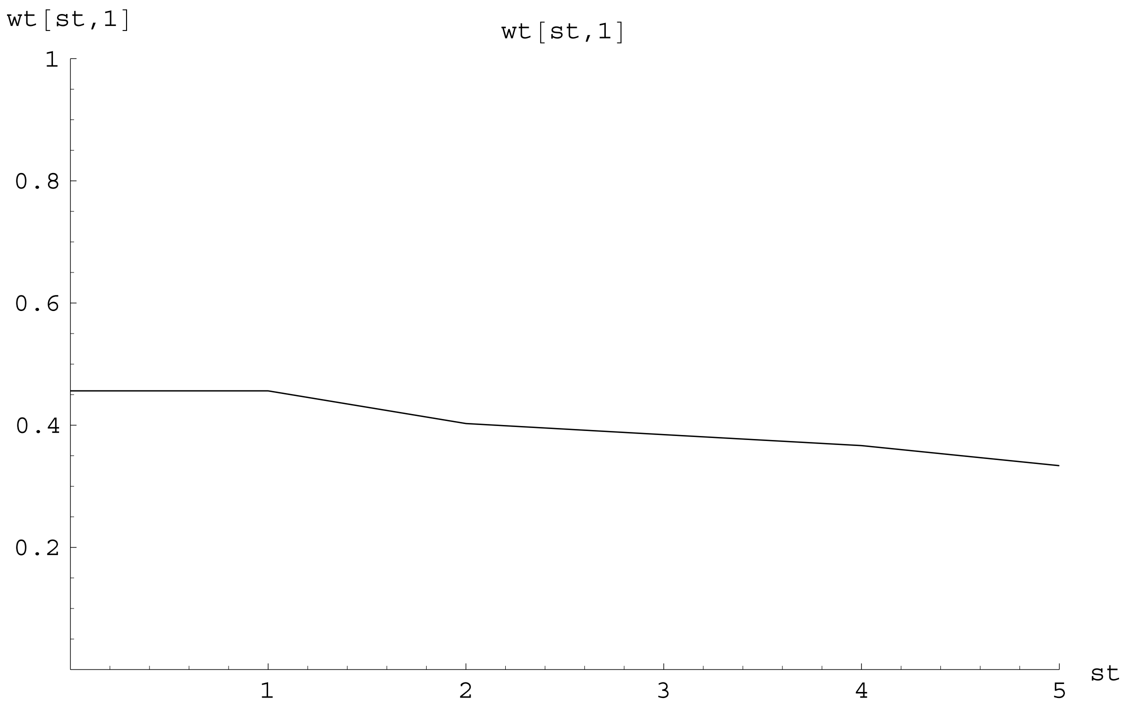

Figure 16:

References

Carroll, C. D. (1996): "Buffer Stock Saving: Some Theory," *Manuscript, Johns Hopkins University, January.*

——— (1997): "Buffer-Stock Saving and the Life Cycle/Permanent Income Hypothesis," *Quarterly Journal of Economics*, CXII(1), 1–56.

Carroll, C. D., and M. S. Kimball (1996): "On the Concavity of the Consumption Function," *Econometrica*, 64(4), 981–992.

Carroll, C. D., J. R. Overland, and D. N. Weil (1998): "Saving and Growth with Habit Formation," *Manuscript, Johns Hopkins University.*