PhD Macroeconomics 1
10. Finite-Horizon Deterministic Dynamic Programming

Karl Whelan

School of Economics, UCD

Autumn 2023

Part I

Introduction to Dynamic Programming

Limitations of Our Existing Models

- The dynamic stochastic models we looked at so far can be used to examine lots of macroeconomic issues but they have limitations. These include
  1. **Infinite Lifetimes**: To allow optimal decision-making rules to be the same for each period, we assumed people (and firms) lived forever. While the infinite horizon assumption might be a useful one on some occasions, it clearly isn't for all cases.
  2. **Treatment of Uncertainty**: We allowed for uncertainty and stochastic shocks but only in a very "smooth" fashion, with random variables being autoregressive processes.
  3. **Constraints**: We did not describe how to handle constraints e.g. limits on how much debt that households or firms are allowed to take on.
  4. **Continuous Decisions**: We modelled agents as making decisions about variables that could take on any values. In some cases, however, it is important to model "all of nothing" decisions (accept a job or not, start an investment project or not).
- We will introduce a technique for solving dynamic models that can address all of these weaknesses. We will develop and solve life-cycle models of consumption and saving in which people have finite lifetimes, also introducing uncertainty and constraints.

Dynamic Programming

- Dynamic programming is a popular method for solving discrete-time dynamic optimisation problems.
- It is an intuitive technique—following the logic of sub-game perfection from game theory—and be easily adapted from deterministic problems to those that incorporate uncertainty.
- It can also solve problems that are not easily solved using the Lagrangian methods we covered previously. For example, it can be applied to problems where the decision variable is not continuous (e.g. an agent deciding whether or not to accept a job) and to problems featuring occasionally binding constraints (e.g. limits on how much people can borrow).
- Let's start with an example of the kind of problem we can use dynamic programming to solve.

$$\underset{\{u_t, x_t\}}{MAX} \sum_{t=1}^{T} F(x_t, u_t, t)$$

subject to

$$x_{t+1} = g(x_t, u_t)$$

x_t is called the **state** variable and u_t is the **control** variable.

The Value Function

- Now define the **value function** of this problem at time t as

$$V_t(x_t) = \overset{MAX}{\{u_k\}_t^T \{x_k\}_{t+1}^T} \sum_{k=t}^{T} F(x_k, u_k, k)$$

- In other words, $V_t(x_t)$ is the value obtained by maximising the sum of the payoffs for the final $T - t + 1$ periods.
- The key insight of dynamic programming is **Bellman's Principle of Optimality** which states that an optimal policy must have the property that, whatever the initial state is, the remaining decisions given this state must constitute an optimal policy. Mathematically, this can be expressed as a recursive equation known as a **Bellman equation**:

$$V_t(x_t) = \overset{MAX}{\{u_t\}} [F(x_t, u_t, t) + V_{t+1}(x_{t+1})]$$

subject to

$$x_{t+1} = g(x_t, u_t)$$

The Value Function with Discounting

- In Economics, usually the function being maximised involves geometric discounting, i.e. we are solving

$$\underset{\{u_t, x_t\}}{MAX} \sum_{t=0}^{T} \beta^t F(x_t, u_t)$$

- In this case, the Bellman equation can adapted to be written as

$$V_t(x_t) = \underset{\{u_t\}}{MAX} \left[F(x_t, u_t) + \beta V_{t+1}(x_{t+1}) \right]$$

subject to

$$x_{t+1} = g(x_t, u_t)$$

Calculating the Value Functions

- The Bellman equation suggests a way to calculate a sequence of value functions based on backward induction.
- Start with the last period, T. In that case, you do not have to worry about the value of the state variable in period $T+1$ and the value function is simply

$$V_T(x_T) = \underset{\{u_T\}}{MAX}\{F(x_T, u_T)\}$$

- Given $V_T(x_T)$, next you calculate

$$V_{T-1}(x_{T-1}) = \underset{\{u_{T-1}\}}{MAX}\left[F(x_{T-1}, u_{T-1}) + \beta V_T(x_T)\right]$$

subject to

$$x_T = g(x_{T-1}, u_{T-1})$$

- Once you have calculated $V_{T-1}(x_{T-1})$, you can then calculate $V_{T-2}(x_{T-2})$ and so on until you have calculated all T value functions.

Calculating Optimal Policies

- Once you have calculated the value functions, you can characterise the optimal $[x_t, u_t]$ path as follows.
- The Bellman Equation is

$$V_t(x_t) = \overset{MAX}{\{u_t\}} \left[F(x_t, u_t) + \beta V_{t+1}(g(x_t, u_t)) \right]$$

- The first-order condition for this problem is

$$\frac{\partial F(x_t, u_t^*)}{\partial u} + \beta V'_{t+1}(g(x_t, u_t^*)) \frac{\partial g(x_t, u_t^*)}{\partial u} = 0$$

- We can also obtain V' by differentiating the Bellman equation and using the envelope theorem (meaning you can ignore terms describing the derivatives of u^*_{t+1} with respect to x because these must equal zero). This gives us:

$$V'_{t+1}(x_{t+1}) = \frac{\partial F(x_{t+1}, u^*_{t+1})}{\partial x} + \beta V'_{t+2}(g(x_{t+1}, u^*_{t+1})) \frac{\partial g(x_{t+1}, u^*_{t+1})}{\partial x}$$

Recursive Optimal Policies

- This gives us two equations

$$\frac{\partial F\left(x_t, u_t^*\right)}{\partial u} + \beta V'_{t+1}\left(g\left(x_t, u_t^*\right)\right)\frac{\partial g\left(x_t, u_t^*\right)}{\partial u} = 0$$

$$V'_{t+1}\left(x_{t+1}\right) = \frac{\partial F\left(x_{t+1}, u_{t+1}^*\right)}{\partial x} + \beta V'_{t+2}\left(g\left(x_{t+1}, u_{t+1}^*\right)\right)\frac{\partial g\left(x_{t+1}, u_{t+1}^*\right)}{\partial x}$$

- From these, we can obtain a set of optimal policies of the form

$$u_t = f\left(x_{t-1}, u_{t-1}\right)$$

  This is usually called the **policy function**.

- This can be combined with

$$x_t = g\left(x_{t-1}, u_{t-1}\right)$$

  to give a recursive solution so that starting from whatever initial conditions we have for the state variables, x_0 we can use these equations to calculate optimal values for both state and control variables.

Part II

An Example: Having Your Cake and Eating It

Optimal Cake Eating

- The problem is the following
  - You're given an initial amount of cake of size X_1 in period one.
  - You've got until period T to eat your cake before health and safety rules dictate that the cake cannot be eaten anymore.
  - You get period-by-period utility of $\log C_t$ from the cake you eat.
  - And you discount utility using discount factor β.
- In other words, your problem is

$$\max_{\{C_t\}_{t=1}^{T}} \sum_{t=1}^{T} \beta^t \log C_t$$

subject to

$$X_{t+1} = X_t - C_t$$

with X_1 given.

Backward Induction: Start At The End

- The Bellman equation for this problem is

$$V_t(X) = \max_C [\log C + \beta V_{t+1}(X - C)]$$

- Let's start at the end. It's period T. It's your last chance to eat the cake so optimal policy is eat all the cake that is left.
- So, entering period T with cake of X_T, the value function is

$$V_T(X_T) = \log X_T$$

- Given this, your problem in period $T-1$ is to maximise

$$V_{T-1}(X_{T-1}) = \max_{C_{T-1}} [\log C_{T-1} + \beta \log(X_{T-1} - C_{T-1})]$$

which implies a first-order condition of the form

$$\frac{1}{C_{T-1}} = \frac{\beta}{X_{T-1} - C_{T-1}}$$

Calculating $V_{T-1}(X_{T-1})$

- This first-order condition can be re-written

$$C_{T-1} = \frac{X_{T-1}}{1+\beta}$$

- Which means we can calculate the value function of period $T-1$ as

$$\begin{aligned} V_{T-1}(X_{T-1}) &= \log\left(\frac{X_{T-1}}{1+\beta}\right) + \beta \log\left(X_{T-1} - \frac{X_{T-1}}{1+\beta}\right) \\ &= \log\left(\frac{X_{T-1}}{1+\beta}\right) + \beta \log\left(\frac{\beta X_{T-1}}{1+\beta}\right) \\ &= (1+\beta)\log X_{T-1} + \beta \log \beta - (1+\beta)\log(1+\beta) \end{aligned}$$

- This can be used to calculate $V_{T-2}(X_{T-2})$.

Calculating $V_{T-2}(X_{T-2})$

- The Bellman equation in period $T-2$ is

$$V_{T-2}(X_{T-2}) = \max_{C_{T-1}} [\log C_{T-2} + \beta V_{T-1}(X_{T-2} - C_{T-2})]$$

which implies a first-order condition of the form

$$\frac{1}{C_{T-2}} = \frac{\beta(1+\beta)}{X_{T-2} - C_{T-2}}$$

- This gives an optimal consumption level of

$$C_{T-2} = \frac{X_{T-2}}{1+\beta+\beta^2}$$

- And $V_{T-2}(X_{T-2})$ turns out to be of the form

$$V_{T-2}(X_{T-2}) = (1+\beta+\beta^2)\log X_{T-2}$$

What Does the Solution Look Like?

- This can be iterated all the way back to the period zero and then the initial consumption level is picked, which determines all the future values. If we had set this problem up as a Lagrangian, we would have obtained a first-order condition of the form

$$\frac{1}{C_t} = \frac{\beta}{C_{t+1}} \Rightarrow C_{t+1} = \beta C_t$$

  which means geometrically declining consumption of cake.

- Initial consumption is set so that the cake is gone after T periods, meaning

$$\left(1 + \beta + \beta^2 + ... + \beta^{T-1}\right) C_1 = X_1$$

  which, using identities about geometric sums simplifies to

$$C_1 = \frac{1-\beta}{1-\beta^{T-1}} X_1$$

- As $T \to \infty$, the solution moves towards $C_t = (1-\beta) X_t$.

HARA Utility and Analytical Solutions

- In some cases, you can obtain analytical solutions for value functions in dynamic programming models.
- For instance, in this case the value functions all took logs of the state variable, the same function as the household had for its period by period utility.
- This property holds if utility is of the Hyperbolic Absolute Risk Aversion (HARA) class i.e.

$$U(C_t) = \frac{1-\gamma}{\gamma}\left(\frac{aC_t}{1-\gamma} + b\right)^b$$

  then the value function has the same functional form as the utility function.
- For example, if the utility function is of the form $U(C_t) = C_t^{\alpha}$, then it turns out that a value function of the form $V_t(K_t) = a_0 + a_1 A^{\alpha}$ will work where A is assets.
- This utility function also generalises to include exponential utility function (set $b = 1$ and $\gamma \to \infty$) and log utility (set $a = 1$ and $\gamma \to 0$).
- In general, however, for most realistic problems in economics, there is no analytical solution to dynamic programming problems and we need to use numerical tools to solve these models.

Part III

Numerical Solution Methods

The Need for Numerical Solution Methods

- While there are occasional dynamic programming problems that have analytical solutions, many of the problems we are interested in do not have such solutions.
- A 1958 review of Bellman's book on dynamic programming in Economica dismissed it by pointing out "*The trouble is that the type of mathematical problem posed in this book is such that it is only under excessively simplified conditions that one can ever hope for a pleasing solution.*"
- Even when analytical solutions exist, it may be very time-consuming to derive analytical solutions for a finite-horizon model with a large T.
- And once we introduce realistic elements like uncertainty and occasionally binding constraints, there are generally no analytical solutions.
- Here, we will introduce numerical dynamic programming.
- Our starting approach will be to work with a slightly modified cake-eating problem, for which we know the analytical solution, and then design and adapt our numerical calculations to see how they match this solution.
- These numerical methods can then be adapted to solve more complex problems.

A Generalised Cake-Eating Problem

- We will use a somewhat generalised version of the cake-eating problem with more general preferences and the asset taking the form of an investment that yields a rate of return.
- The problem is the following
  - You're given an initial stock of assets of value A_1 in period one.
  - You've got until period T to eat your cake before health and safety rules dictate that the cake cannot be eaten anymore.
  - You get period-by-period utility of $\frac{C_t^{1-\gamma}}{1-\gamma}$ from the cake you eat.
  - And you discount utility using discount factor β.
- In other words, your problem is

$$\max_{\{C_t\}_{t=1}^{T}} \sum_{t=1}^{T} \beta^t \frac{C_t^{1-\gamma}}{1-\gamma}$$

subject to

$$A_{t+1} = (1+r)(A_t - C_t)$$

with A_1 given.

The Analytical Solution

- Let
$$\alpha = \beta^{\frac{1}{\gamma}} (1+r)^{\frac{1-\gamma}{\gamma}}$$
- Then the analytical solution for the optimal path for consumption for this problem is
$$C_t = \left(\frac{1-\alpha}{1-\alpha^T} \right) (\beta (1+r))^{\frac{t-1}{\gamma}} A_1$$
- Knowing this, we can calculate the optimal path for assets using the identity subject to
$$A_{t+1} = (1+r)(A_t - C_t)$$
with A_1 given.
- Note that consumption can either increase or decrease over time depending on whether $\beta (1+r)$ is greater than or less than one. If it equals one (as in the Hall random walk model of consumption) then consumption will be constant for the entire time.
- We will now work on a numerical method for solving this problem and assess it by comparing the answers it generates to the known true solution.

Numerical Approximation Methods

- We want our numerical method to start in period T and calculate the value function of the state variable, which is assets on hand, $V^T(A_T)$.
- Then, we want it to work backward in time to calculate all of the other $V^t(A_t)$ functions, from which we can calculate optimal behaviour.
- In theory, we need to find out the value of $V^t(A_t)$ for every possible value of assets.
- In practice, computers can't evaluate functions at every single feasible real numbered value for assets so instead we pick a number of points on a grid and make assumptions about the behaviour of the value function at points in between.
- For example, you could assume the function is piece-wise linear between the selected grid point.
- We will start with a very simple grid-based approach to solving a simple version of this model with no labour income and no constraints on borrowing.
- Later we will refine our computational method and discuss further possible refinements.

A Simple Grid-Based Approach

- We will now discuss a Matlab programme that takes a simple approach to solving this model using numerical methods.
- Rather than try to calculate what the value functions equal for all possible values for assets on hand, we only look at points on a grid, going from zero to a multiple of the starting value for assets.
- We normalise starting assets equal to one and, in this case, because we have calibrated households to be "impatient" ($r = 0.01$, $\beta = 0.95$), we know the solution always involves falling assets, so we don't consider values for assets on hand higher than the starting level.
- When you chart your solution, it will be clear whether your choice of a restricted grid has affected the solution—you see periods where assets on hand consistently equal the maximum or minimum amount you have allowed. You can go back and reset the maximum or minimum asset values so they don't constrain the households.
- The preliminaries in the programme set the coefficients and initialise the various things we will be calculating (value functions, policy functions, final choices of optimal consumption and assets) equal to zero.

Setting Up the Programme

```
% 1. Preliminaries
tic;        % start the clock
clear all;
close all;
clc;
global gamma
```

```
%% 2. Parameter Values
T = 60 ;  % Finite Horizon
time = linspace(1,T+1,T+1);
ngrid = 300;                      % number of data points in the grid
gamma = 1.5 ;                      % CRRA utility parameter
r     = 0.01;                      % Interest rate
beta = 0.95;                       % discount factor
StartingA = 1.5;                   % Starting amount of assets
minCons = 1e-5;                    % min allowed consumption
```

```
%%
% 3. Deciding how high to let the asset grid go relative to starting assets
Maxassetratio = 1;
amin          = 0 ;
amax          = StartingA*Maxassetratio;    % Top of asset grid is a multiple of the starting assets
Index         = NaN(T+1,1);
Index(1,1)    = ngrid / Maxassetratio ;     % First entry of Index is adjusted to have correct starting value
```

```
%%
% 4. Initialising matrices
V        = NaN(T, ngrid);
PolicyA  = NaN(T, ngrid);
PolicyC  = NaN(T, ngrid);
IndexA   = NaN(T, ngrid);

Agrid         = linspace(amin,amax,ngrid);
OptimalA      = NaN(T+1,1);
OptimalC      = NaN(T+1,1);
AnalyticalA   = NaN(T+1,1);
AnalyticalC   = NaN(T+1,1);
Cons_options  = NaN(ngrid, 1);
Value_options = NaN(ngrid, 1);
```

Solving for the Final Period. Then Working Backwards

- With preliminaries set, the programme then calculates the last period's value function.
- This is easy: The "consumption policy" is to set consumption equal to assets on hand so the value function equals the utility function evaluated at assets on hand. (I have specified the utility function in a separate programme).
- The programme then uses a loop to gradually work backwards over time to calculate the rest of the value functions.
- For each period, the programme looks at the options for this period's consumption and next period's assets, given a fixed value of this period's assets.
- The lower consumption is, the higher assets will be next period. Negative consumption is not allowed, so values of assets that are infeasible are not considered when calculating optimal policy.
- For each level of assets on hand, the optimal consumption value (and thus the optimal value of next period's assets) is calculated via applying the Bellman equation.
- With policy rules solved, we then solve forward from the initial level of assets.

Solving for the Final Period. Then Working Backwards

```
%% 5. Final Period
% Set the terminal value function to consuming all assets
for i=1:ngrid
V(T,i) = utility(Agrid(i)) ;
PolicyC(T,i) = Agrid(i);
end
```

```
%%
% 6. Working backwards in time to calculate optimal policy functions
for t=T-1:-1:1
for i=1:ngrid % Looping over current period assets
Cons_options  = Agrid(i) - Agrid / (1+r);
Cons_options(Cons_options <0) = NaN;  % Not allowing negative values of consumption
VT1 = V(t+1,:);

Value_options = utility(Cons_options) + beta*VT1;
[max_val,idx] = max(Value_options);
V(t,i)        = max_val;
IndexA(t,i)   = idx;
PolicyA(t,i)  = Agrid(idx);
PolicyC(t,i)  = Cons_options(idx);
end % i loop

fprintf('Solved period %d of %d.\n',t, T)
end %t loop
```

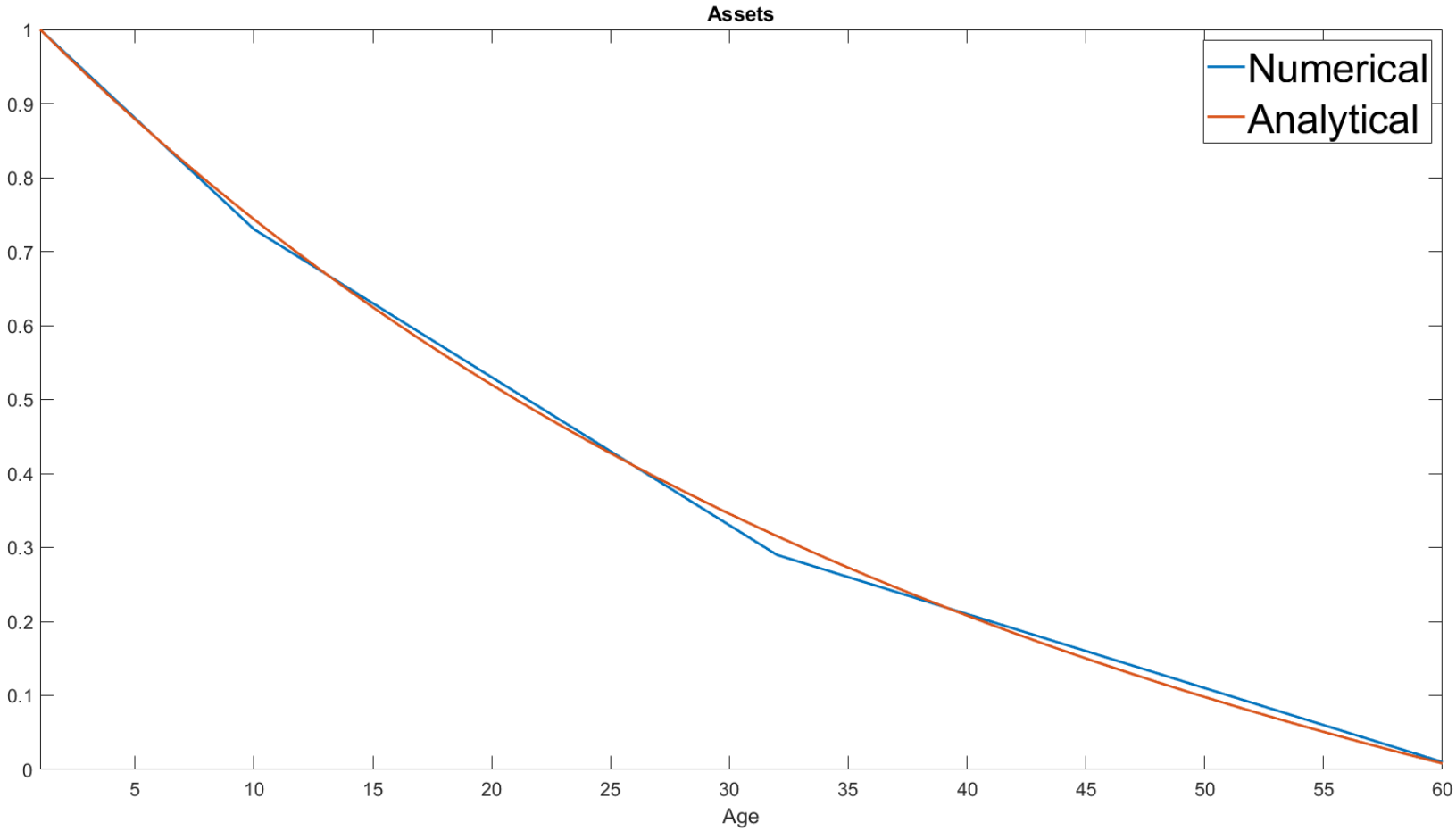
Solving Forward from Initial Capital Using the Policy Rules

```
%%
% 7. Moving forward from the first period to calculate optimal path for
% assets and consumption

% First calculate the sequence of optimal points on the grid, starting from
% our % initial value for assets.
for t=2:T
Index(t) = IndexA(t-1,Index(t-1));
end
Index(T+1) = 0;

% Then translate the optimal points into asset values from the policy
% matrix
OptimalA(1) = StartingA;
for t=2:T
OptimalA(t) = PolicyA(t-1,Index(t-1));
end
OptimalA(T+1) = 0;

% Then calculate the implied values of consumption
for t=1:T-1
OptimalC(t) = OptimalA(t) -OptimalA(t+1)/(1+r) ;
end
OptimalC(T) = OptimalA(T);
OptimalC(T+1) = 0;
```

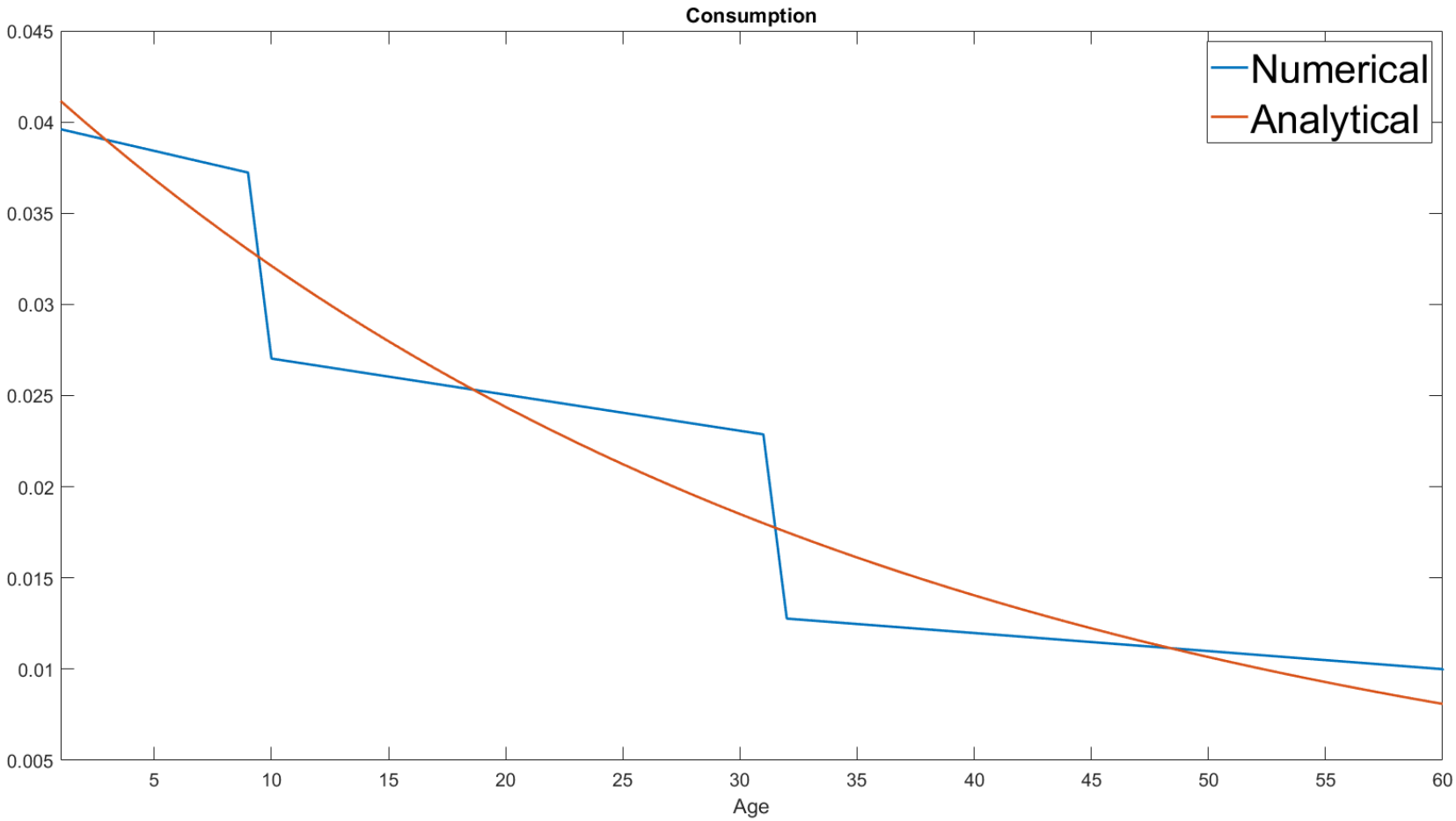
Solution Quality Depends on Grid Size

- How well does our numerical solution method do?
- We started with a grid of size 100. We see that it approximates the true analytical solution for assets quite well.
- Despite this, the chart for consumption (which is clearly on a different scale to the one for assets) shows our solution for consumption is much more "jerky" than the true solution.
- The problem is we are only evaluating assets for a limited number of possible values and so the "best" consumption levels that we find are not really the best, just the best given the limited set of possible asset values we provided.
- The following pages show how this problem can be fixed by specifying a finer and finer grid, with more and more possible values for assets.
- By the time the asset grid has 5000 values, the numerical solution and analytical solution for consumption are essentially the same.

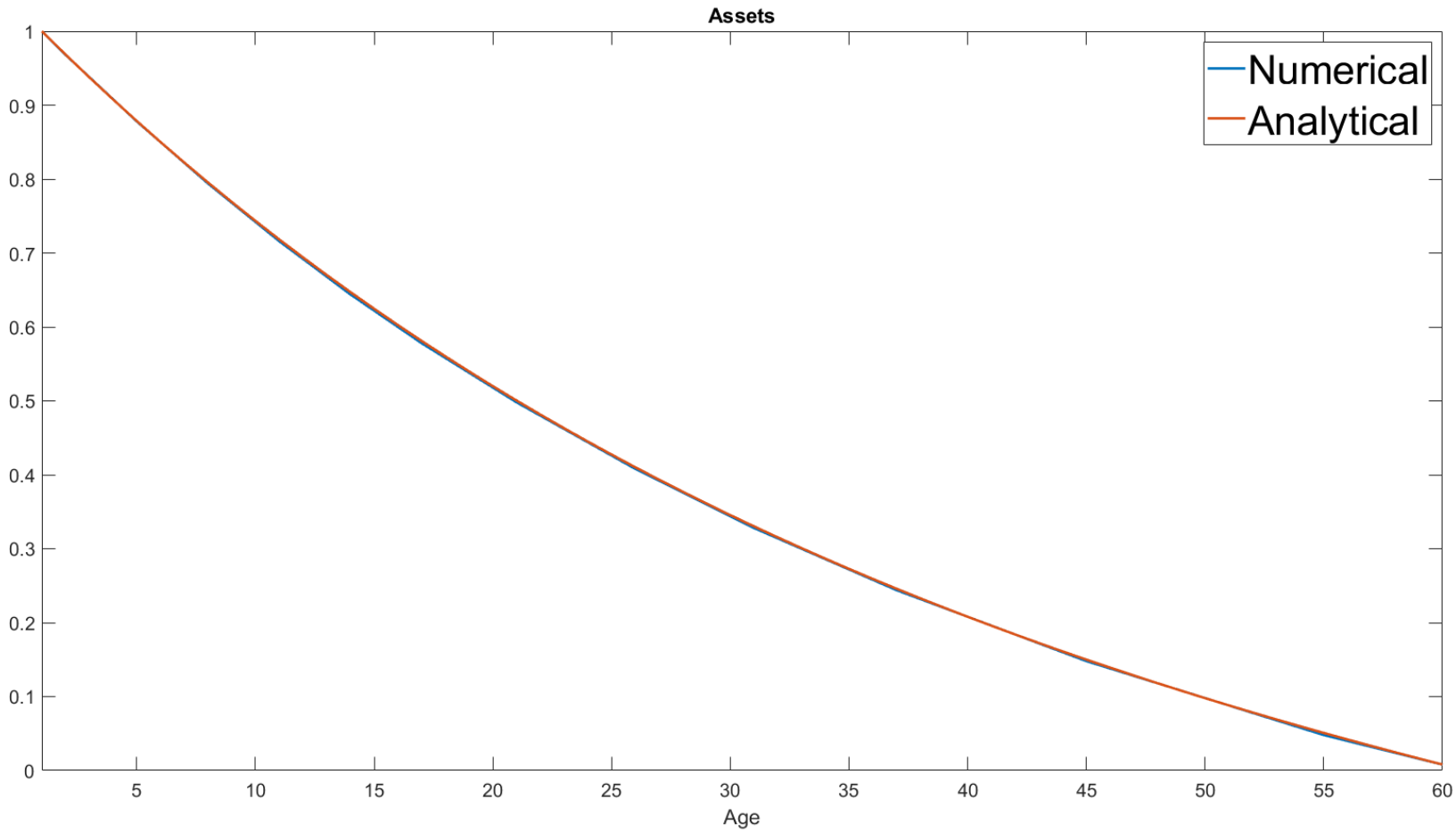
Actual Assets versus Analytical: Grid of Size 100



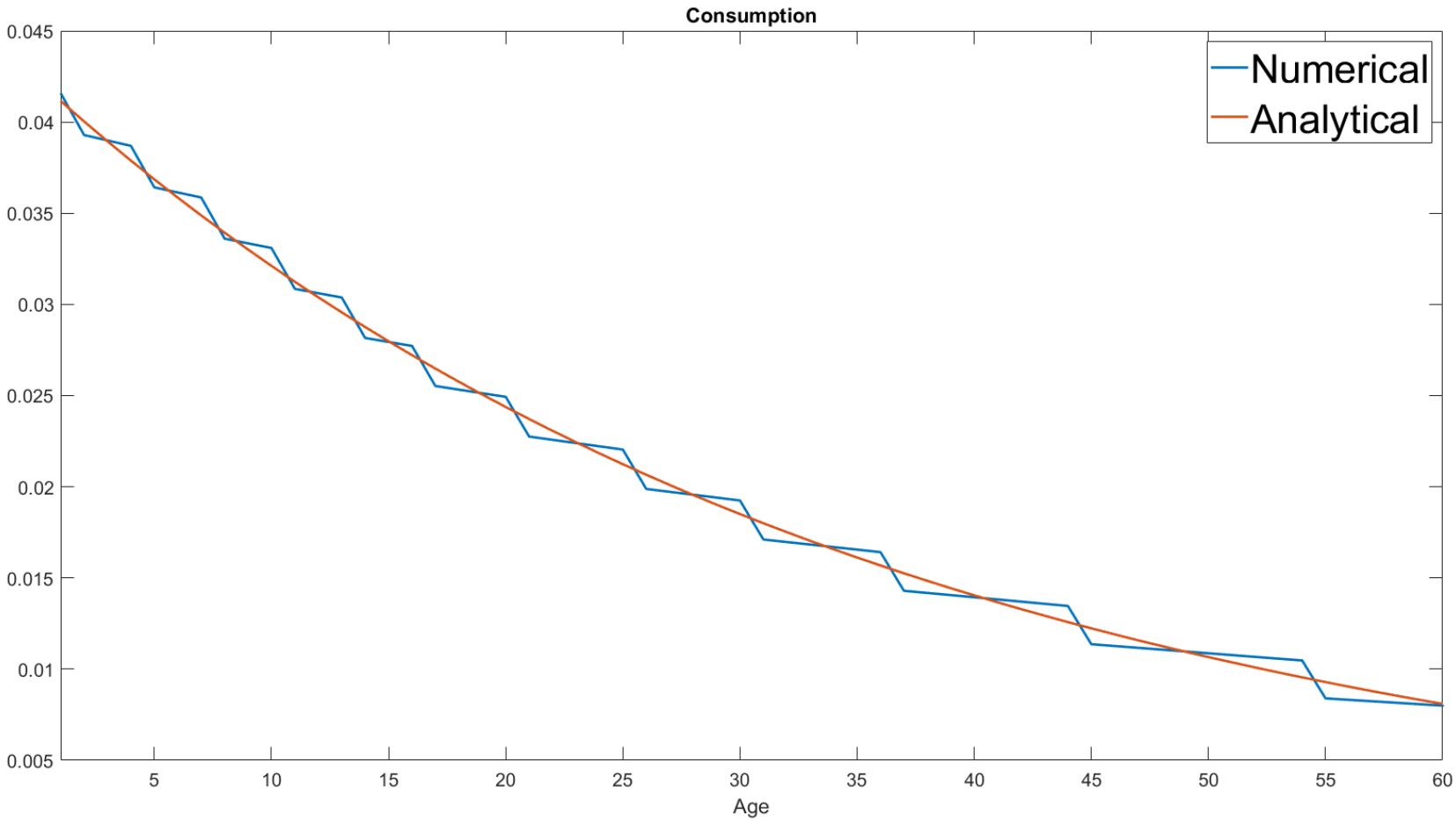
Actual Consumption versus Analytical: Grid of Size 100



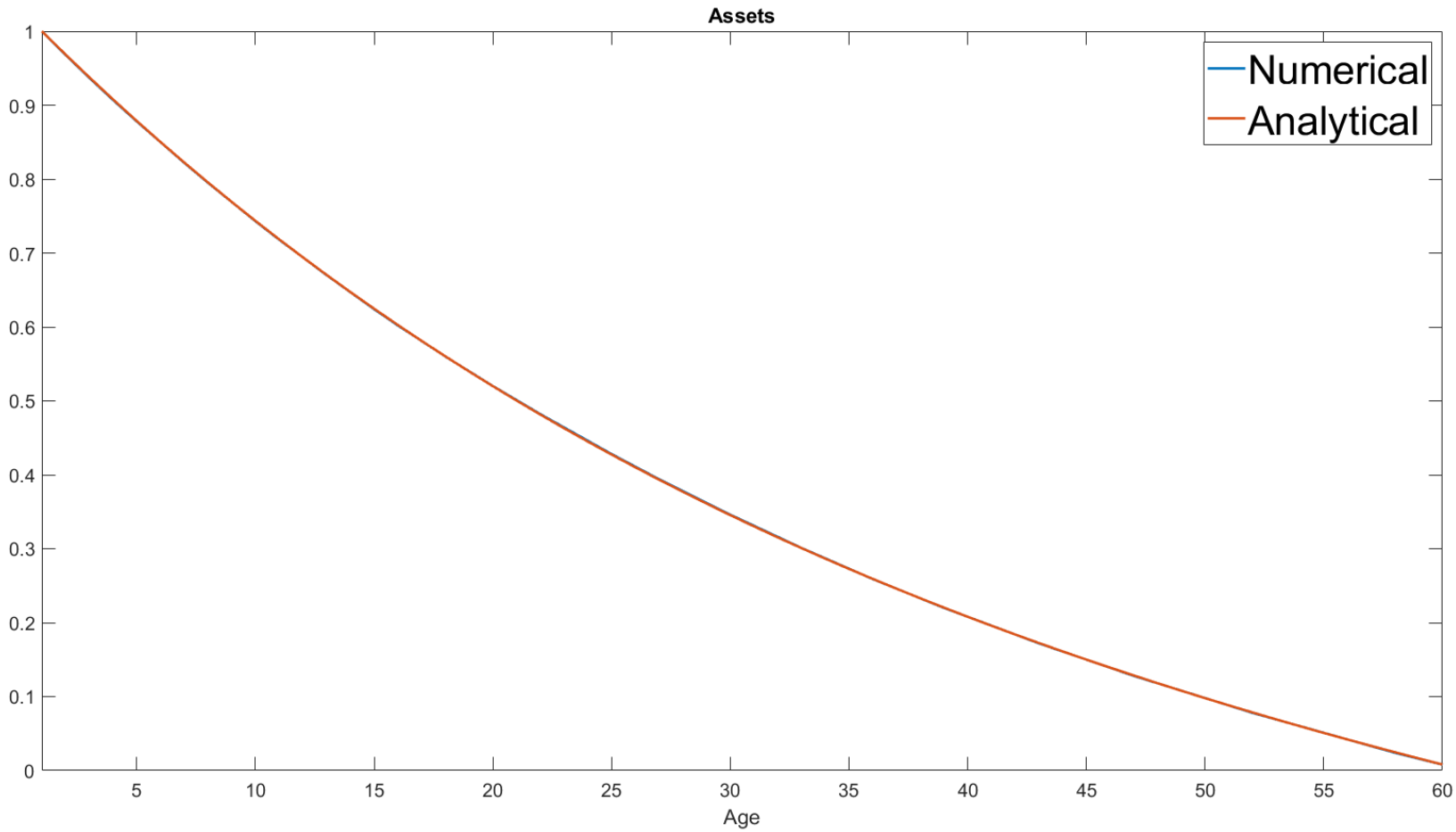
Actual Assets versus Analytical: Grid of Size 500



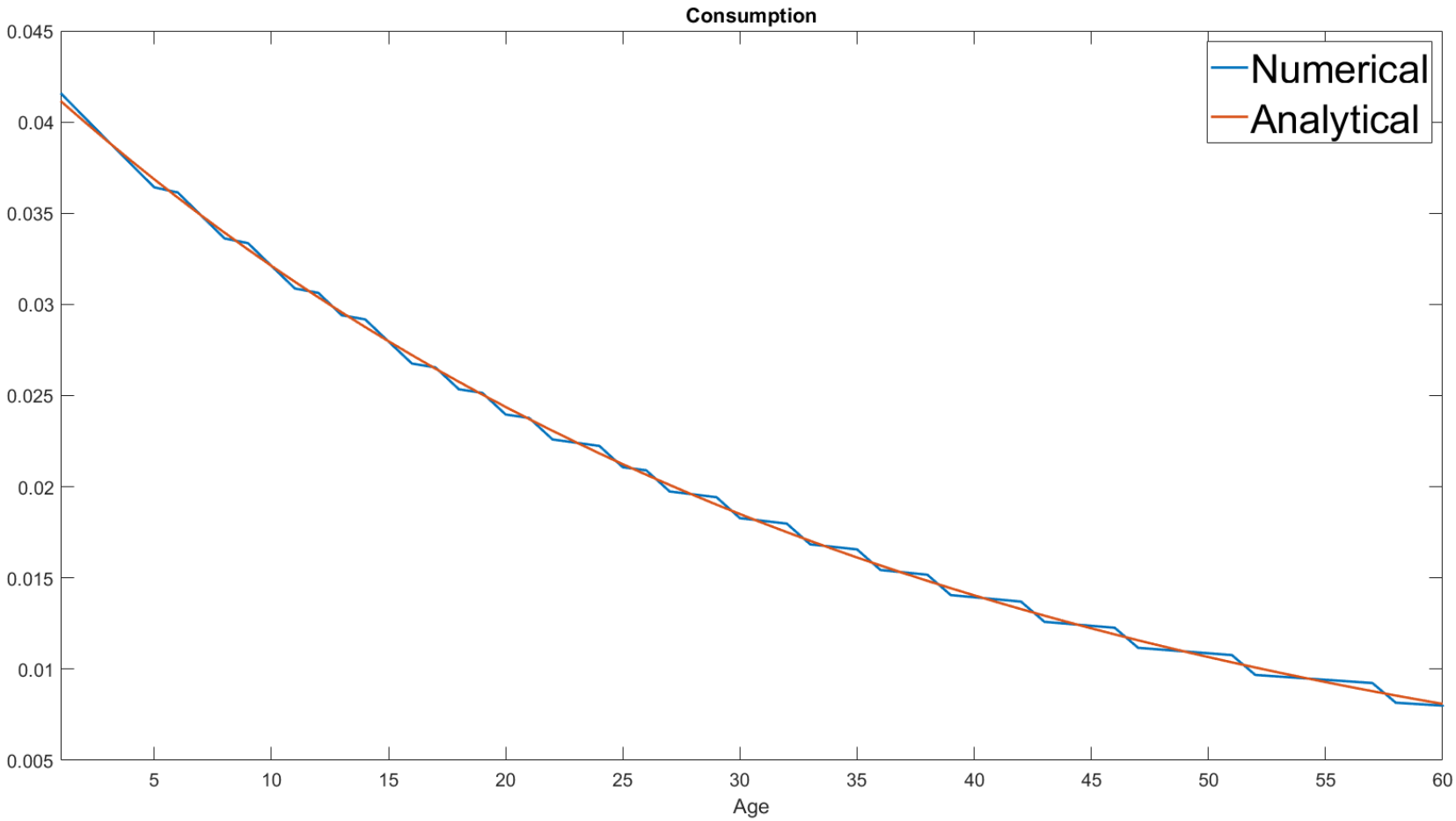
Actual Consumption versus Analytical: Grid of Size 500



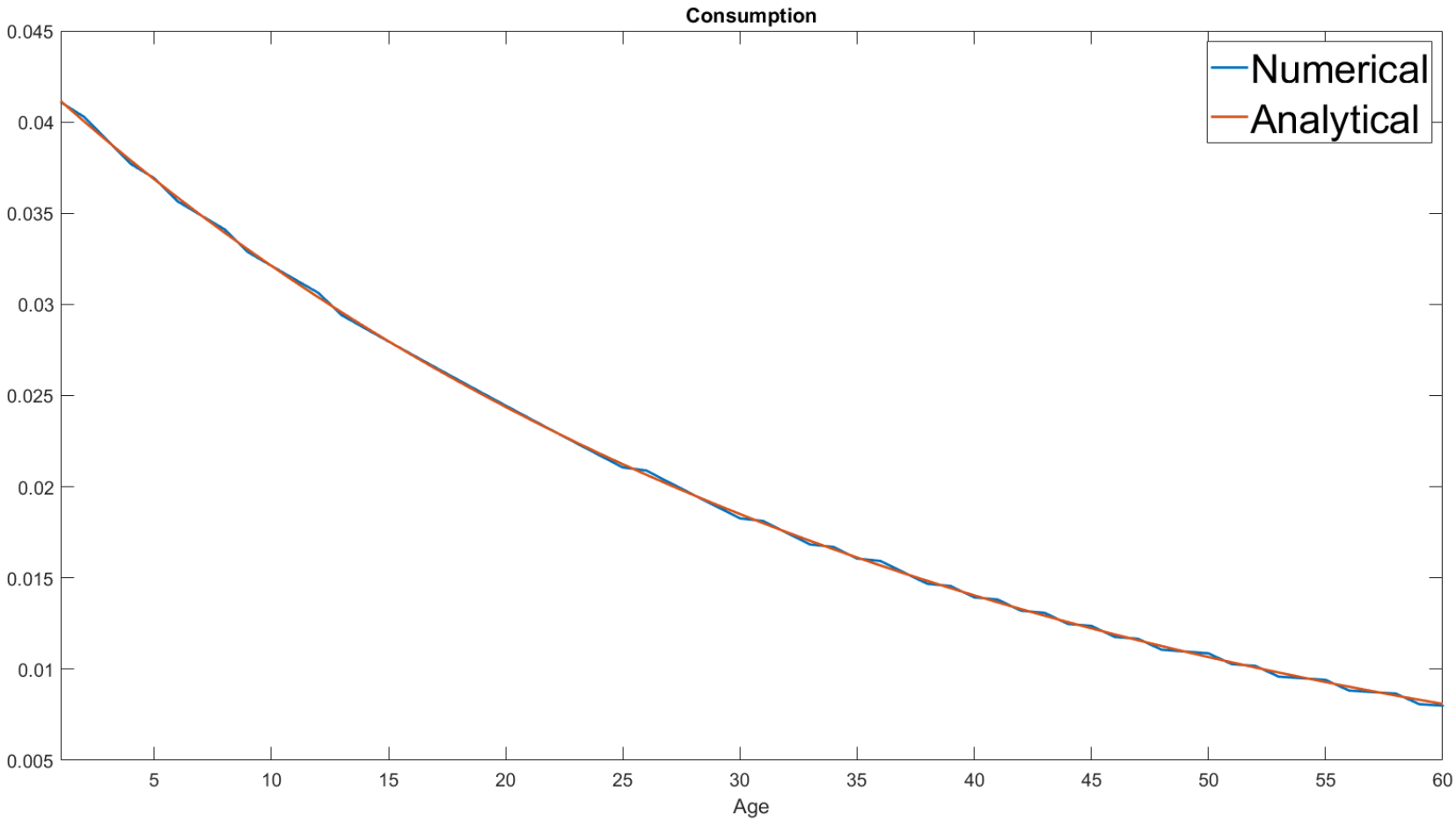
Actual Assets versus Analytical: Grid of Size 1000



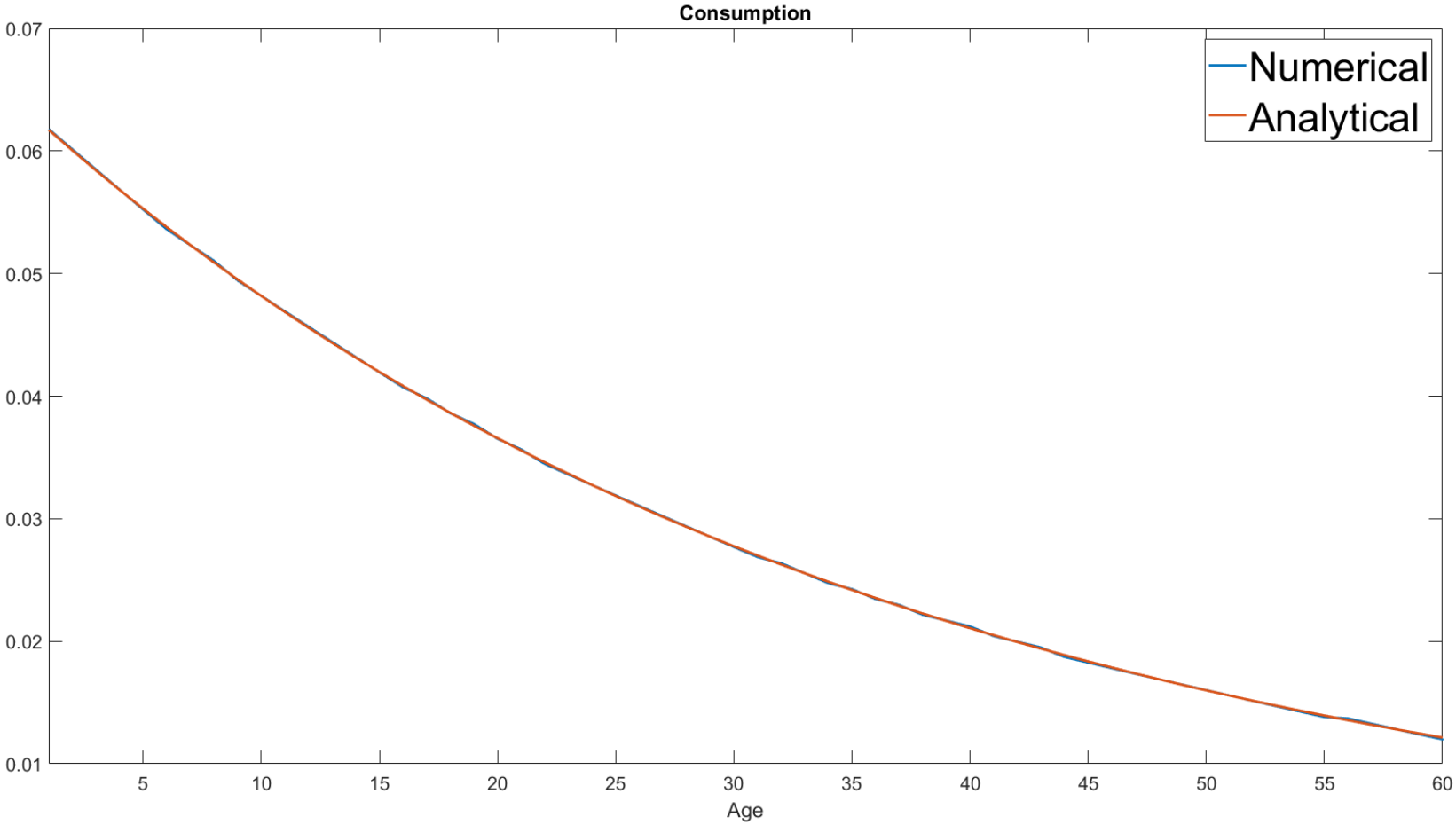
Actual Consumption versus Analytical: Grid of Size 1000



Actual Consumption versus Analytical: Grid of Size 2000



Actual Consumption versus Analytical: Grid of Size 5000



Two Improvements Using Interpolation

- The previous programme works ok but we can improve speed and accuracy.
- One problem with large grids is it takes longer for the programme to solve. This isn't that big a deal for this specific problem. The chart on the previous page took my (admittedly fairly fast) computer just under a minute to solve.
- However, when we move on to introducing uncertainty, we will have to perform these kind of calculations many many times so solving this kind of grid size will take a really long time.
- Our next programme uses two different grids, one small ($n = 300$) and one big ($n = 5000$). We use the smaller grid to consider starting assets for each period but then use the bigger grid to allow for more precise calculation of the optimal value of consumption. This is much quicker but doesn't lose much accuracy.
- We also improve accuracy by not using the restricted consumption policy functions calculated over the 300 asset grid points but instead linear interpolated version of these functions that can be evaluated at any feasible value for assets on hand, not just those on the grid. This programme is almost 20 times faster and produces essentially the same output

Two Grids. Need to Adjust Size of Some of the Matrices

```
% 1. Preliminaries
tic;        % start the clock
clear all;
close all; clc;
global gamma
```

```
%% 2. Parameter Values
T         = 60 ;  % Finite Horizon
time      = linspace(1,T+1,T+1);
gamma     = 1.5;                    % CRRA utility parameter
r         = 0.03;                   % Interest rate
beta      = 0.95;                   % discount factor
StartingA = 1;                      % Starting amount of assets
minCons   = 1e-5;                   % min allowed consumption
```

```
%%
% 3. Deciding how high to let the asset grid go relative to starting assets
StartingA     = 1;
Maxassetratio = 1;
ngrid1        = 300;
ngrid2        = 5000;
amin          = 0;
amax          = StartingA*Maxassetratio;
agrid         = linspace(amin,amax,ngrid1);
agrid2        = linspace(amin,amax,ngrid2);
```

```
%%
% 4. Initialising matrices
VFinal    = NaN(1,ngrid1);
PolicyA   = NaN(T, ngrid1);
PolicyC   = NaN(T, ngrid1);
IndexA    = NaN(T, ngrid1);
TV        = NaN(T, ngrid1);

Vtk          = NaN(1,ngrid1);
OptimalA     = NaN(T+1,1);
OptimalC     = NaN(T+1,1);
AnalyticalA     = NaN(T+1,1);
AnalyticalC     = NaN(T+1,1);

V             = NaN(T, ngrid2);
Cons_options  = NaN(ngrid2, 1);
Value_options = NaN(ngrid2, 1);
```

Using Interpolation to Allow More Precise Policy Functions

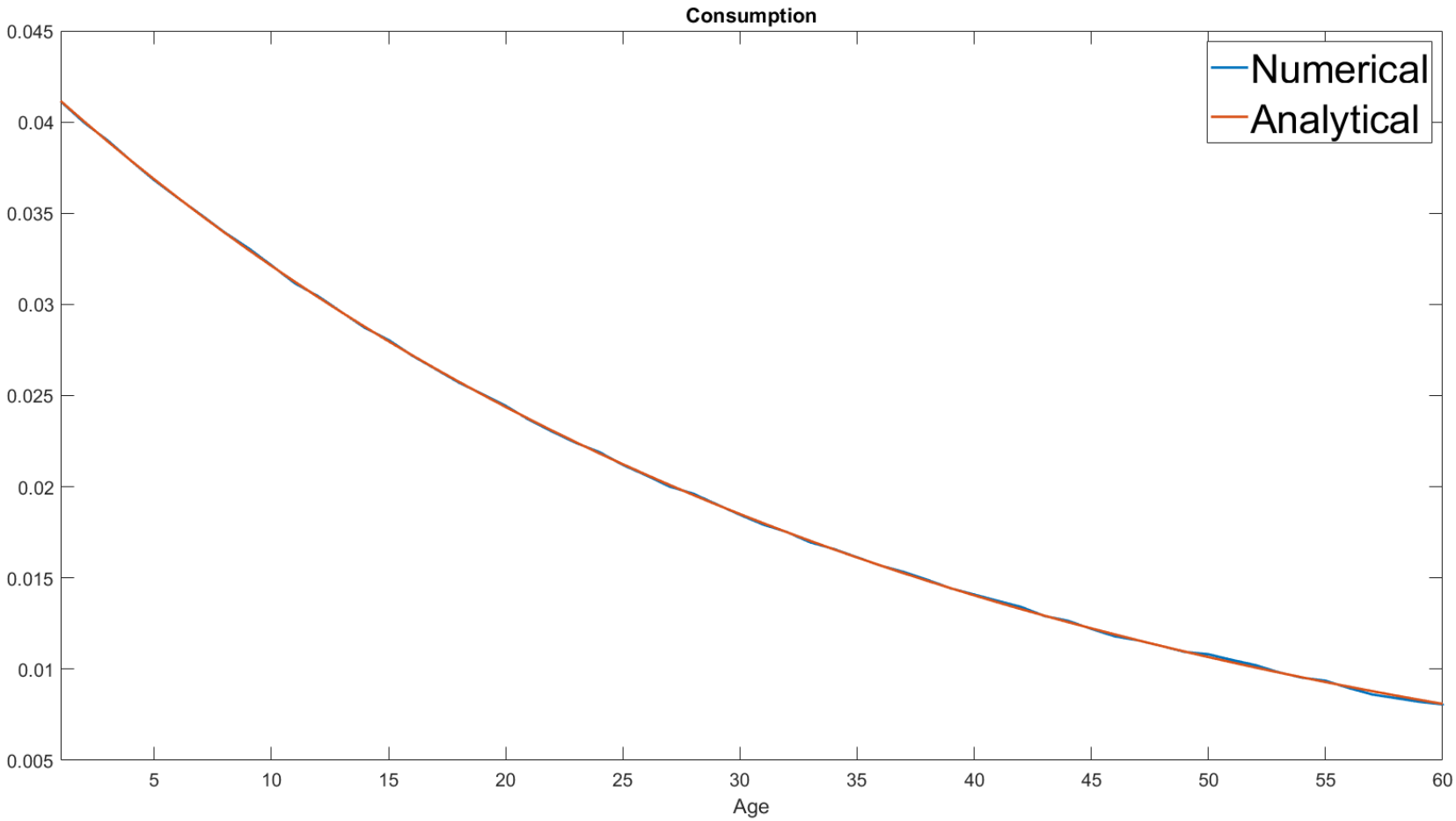
```
%% 5. Final Period
% Set the terminal value function to consuming all assets
for i=1:ngrid1
VFinal(i)  = utility(agrid(i)) ;
PolicyC(T,i) = agrid(i);
end

% Interpolating the final-period value function over a finer grid
V(T,:) = interp1(agrid,VFinal(:),agrid2,'linear','extrap');
```

```
%%
% 6. Working backwards in time to calculate optimal policy functions
for t=T-1:-1:1
for i=1:ngrid1 % Looping over current period assets
Cons_options  = agrid(i) - agrid2 / (1+r);
Cons_options(Cons_options <0) = NaN;  % Not allowing negative values of consumption
VT1 = V(t+1,:);

Value_options = utility(Cons_options) + beta*VT1;
[max_val,idx] = max(Value_options);
TV(t,i)       = max_val;
IndexA(t,i)   = idx;
PolicyA(t,i)  = agrid2(idx);
PolicyC(t,i)  = Cons_options(idx);
end % i loop

% Again interpolating the value function over a finer grid
V(t,:) = interp1(agrid,TV(t,:),agrid2,'linear','extrap');

fprintf('Solved period %d of %d.\n',t, T)
end %t loop
```

Solving Forward Using the Interpolated Policy Functions

```
%%
% 7. Moving forward from the first period to calculate optimal path for assets and consumption
OptimalA(1) = StartingA;
for t=1:T-1
    PC = PolicyC(t,:);
    OptimalC(t) = max(0, interp1(agrid, PC, OptimalA(t),'linear','extrap') );
    OptimalA(t+1) = (1+r)*(OptimalA(t) - OptimalC(t));
end

OptimalC(T,1) = OptimalA(T);
OptimalC(T+1,1) = 0;
OptimalA(T+1,1) = 0;
```

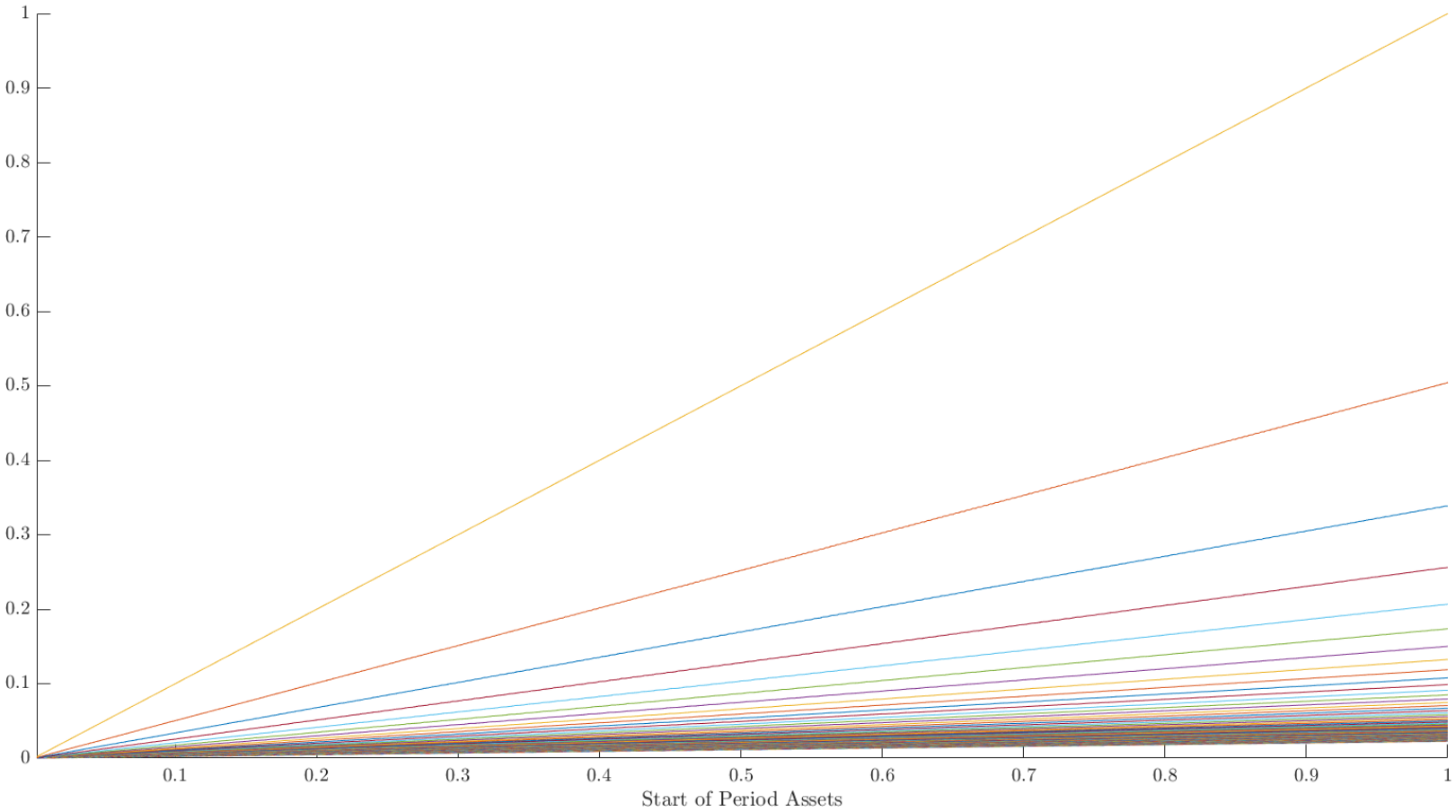
Actual Consumption versus Analytical: Double Grid $(300, 5000)$ and Continuous Policy Function



Optimal Consumption Rules At Various Stages

- This model is deterministic, so given starting assets, there is a unique optimal solution for what plays out over time.
- However, we have calculated optimal consumption rules for each point in time dependent on how much assets on hand equal at the start of the period and it is worth looking at these rules.
- In the final year of life, optimal strategy is to consumer whatever assets are left. We also see a number of other lines corresponding to a few years before the end of life when the propensity to consume from assets is also very high.
- However, for most of life, the MPC from assets is fairly low. This suggests that "infinite horizon" models may do ok at describing consumption patterns for most of life but will not do a good job in the final years.

Optimal Consumption Rules at Various Ages



Part IV

A Life-Cycle Model of Consumption

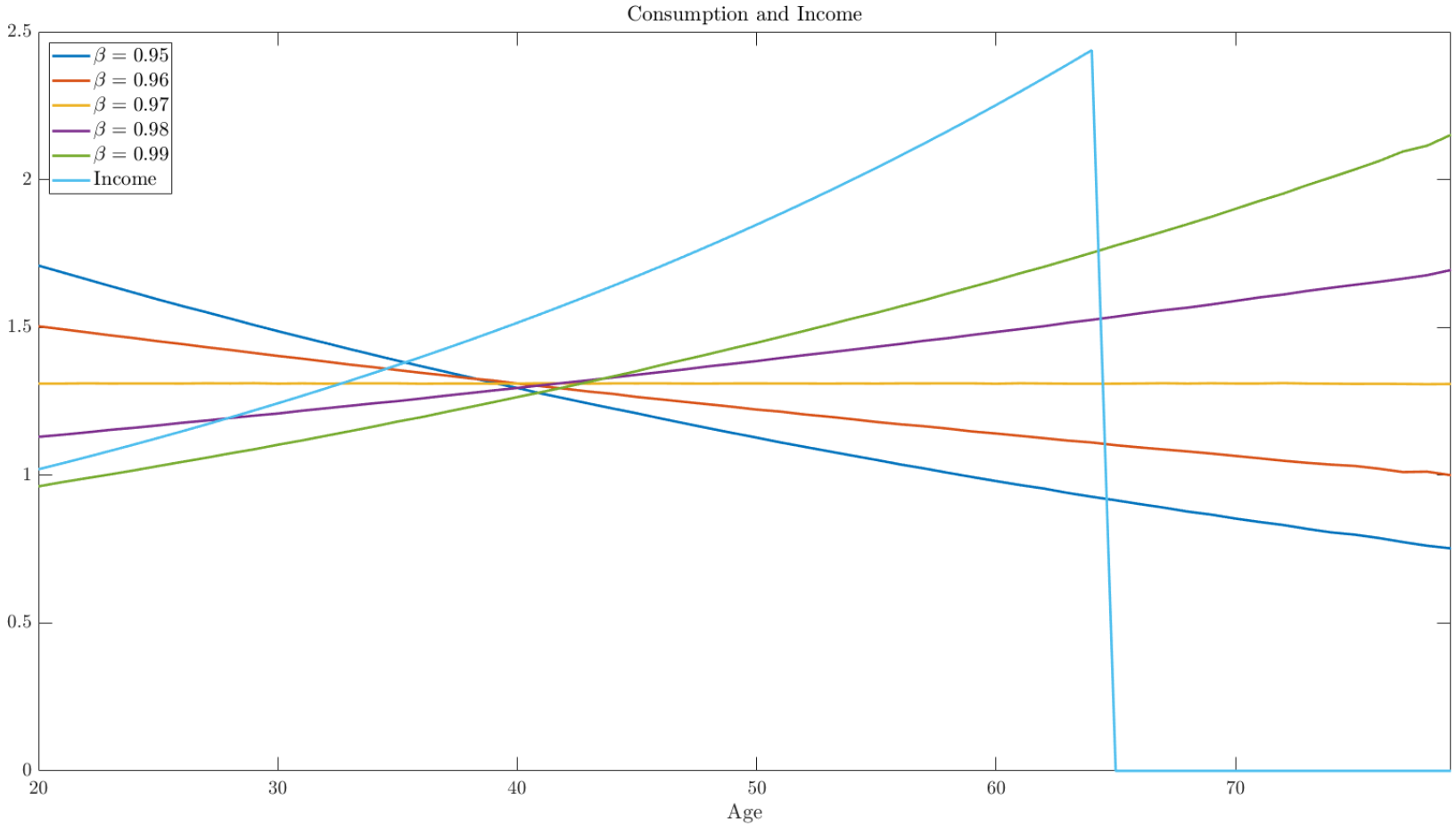
Specifying the Model

- A realistic model of life-cycle consumption needs to account for two important patterns: People earn labour income and then retire.
- Our life-cycle model has the following features:
  - People spend their first 19 years as children and in education.
  - The problem thus starts at age 20 and we will keep $T = 60$, representing a 60-year life span from the time of starting work.
  - People start their working life with no assets (no inherited wealth).
  - They then work for 45 years, with initial labour income of 1, which then grows at 2 percent per year until they retire.
  - People retire at age 65 (thus earning no more labour income) and then live until they are 80 with only their assets to fund consumption.
  - In the first version of the model, people can borrow as much as they want. In the second version, they are not allowed to have a negative net asset position.
- Obviously, the idea that people know exactly what their lifetime income is going to be is not an accurate assumption. And knowing exactly when you will die (and not having to worry about medical expenses) are also issues. We will discuss integrating uncertainty into this kind of model next week.

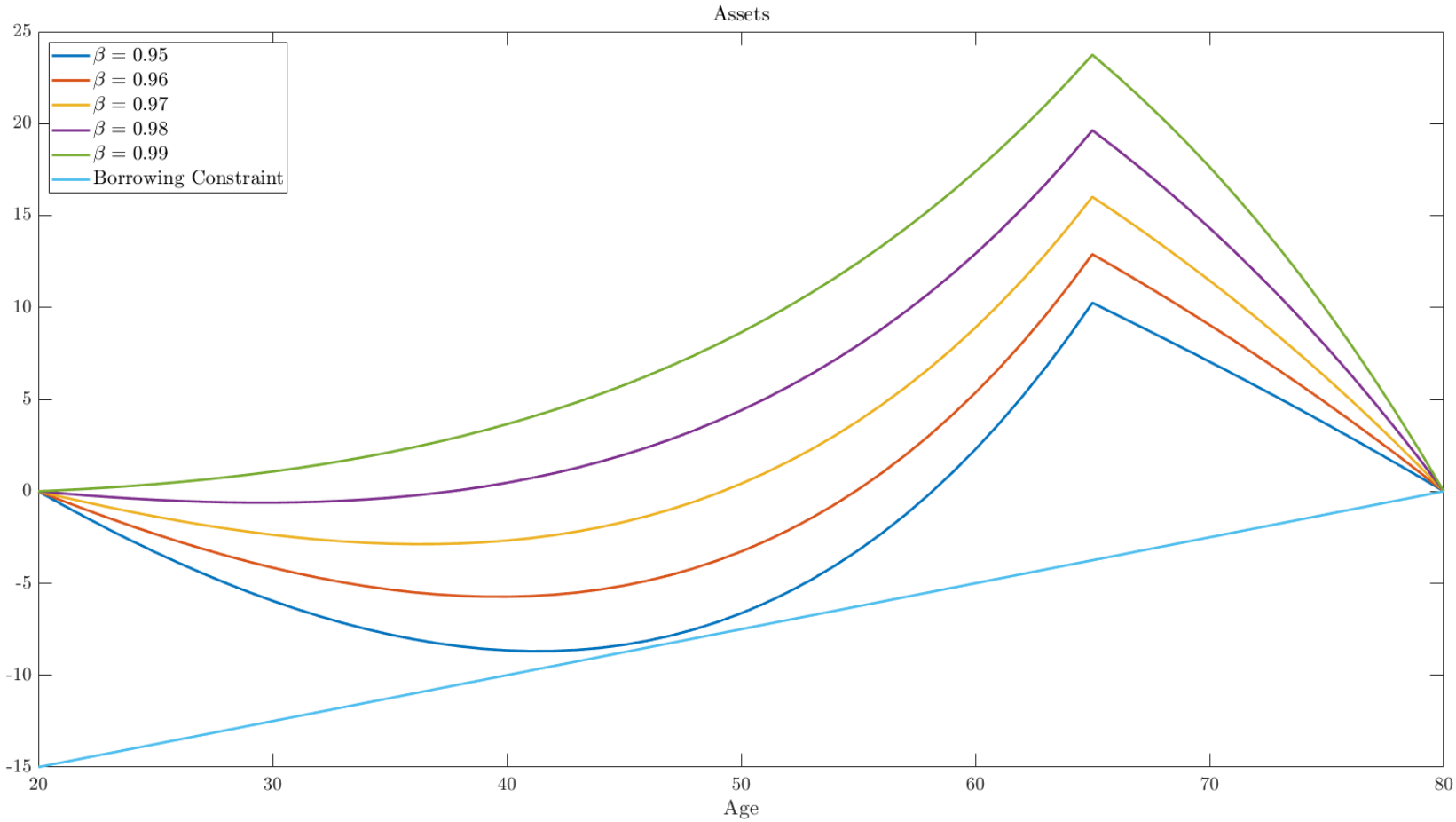
Running the Model with Patient and Impatient Households

- The charts on the next few pages run our model for various values of β.
- We set $r = 1/0.97 - 1$, so that the consumers with $\beta = 0.97$ want to perfectly smooth their consumption because $\beta(1 + r) = 1$.
- We start by letting people borrow as much as they want early in life and then paying off these debts later.
- This was implemented by allowing people to borrow a lot at the start of life and then (in a linear fashion) tightening the borrowing constraint so they needed to end life with no borrowings. By setting this initial amount of possible borrowing high enough, we can get a set of simulations in which the borrowing constraint doesn't limit behaviour early in life.
- Patient households (e.g. $\beta = 0.99$) never borrow and build up a big stock of assets to sustain growing consumption over life.
- "Random walk" households ($\beta = 0.97$) borrow initially and then run up a smaller stock of assets to allow constant consumption over life.
- Patient households (e.g. $\beta = 0.95$) borrow a lot early in life and build up a small stock of assets and have falling consumption over life.

Life Cycle Model: Consumption Paths with Negative Net Assets Allowed ($r = 1/0.97 - 1$)



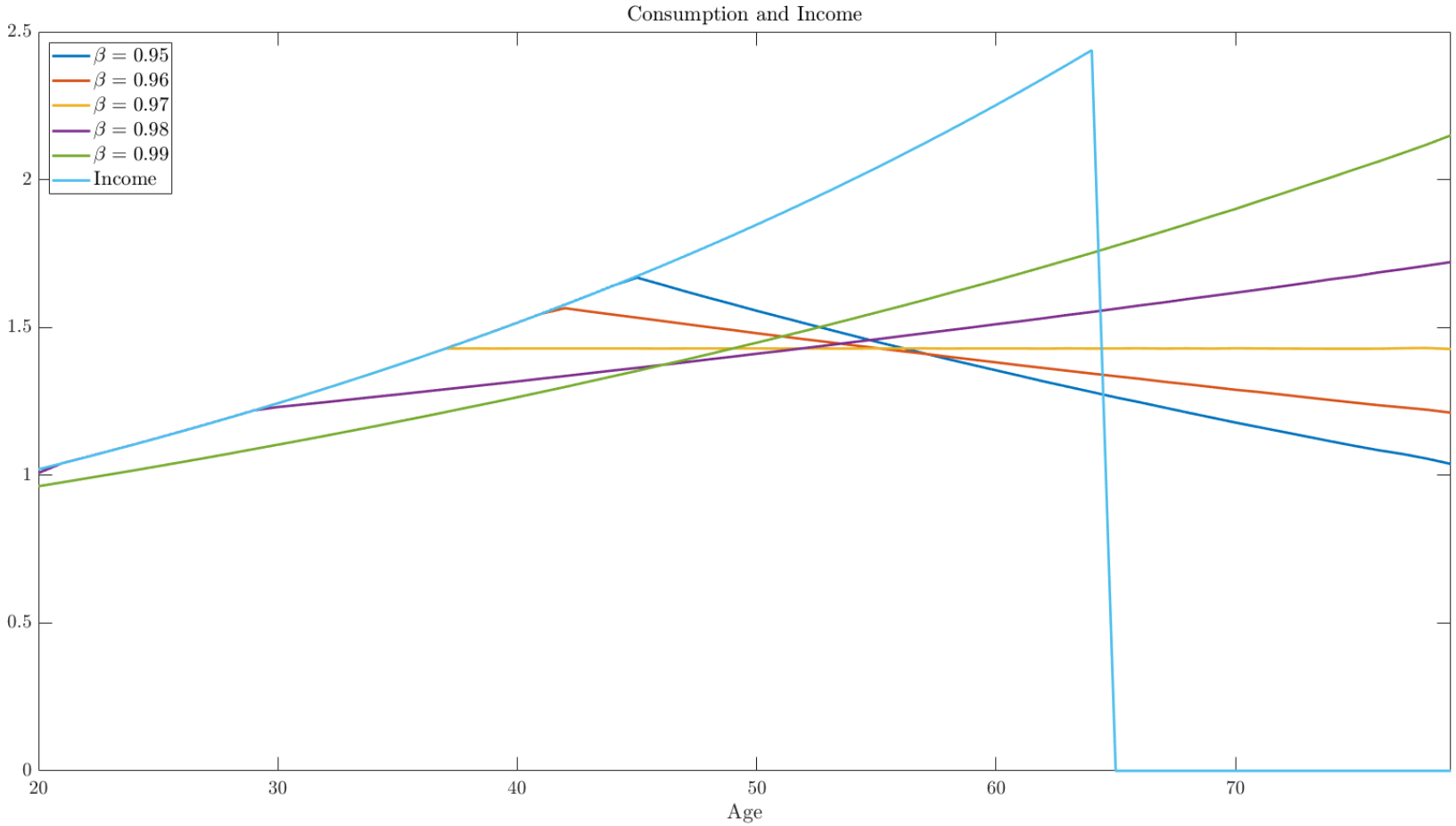
Life Cycle Model: Asset Paths with Negative Net Assets Allowed ($r = 1/0.97 - 1$)



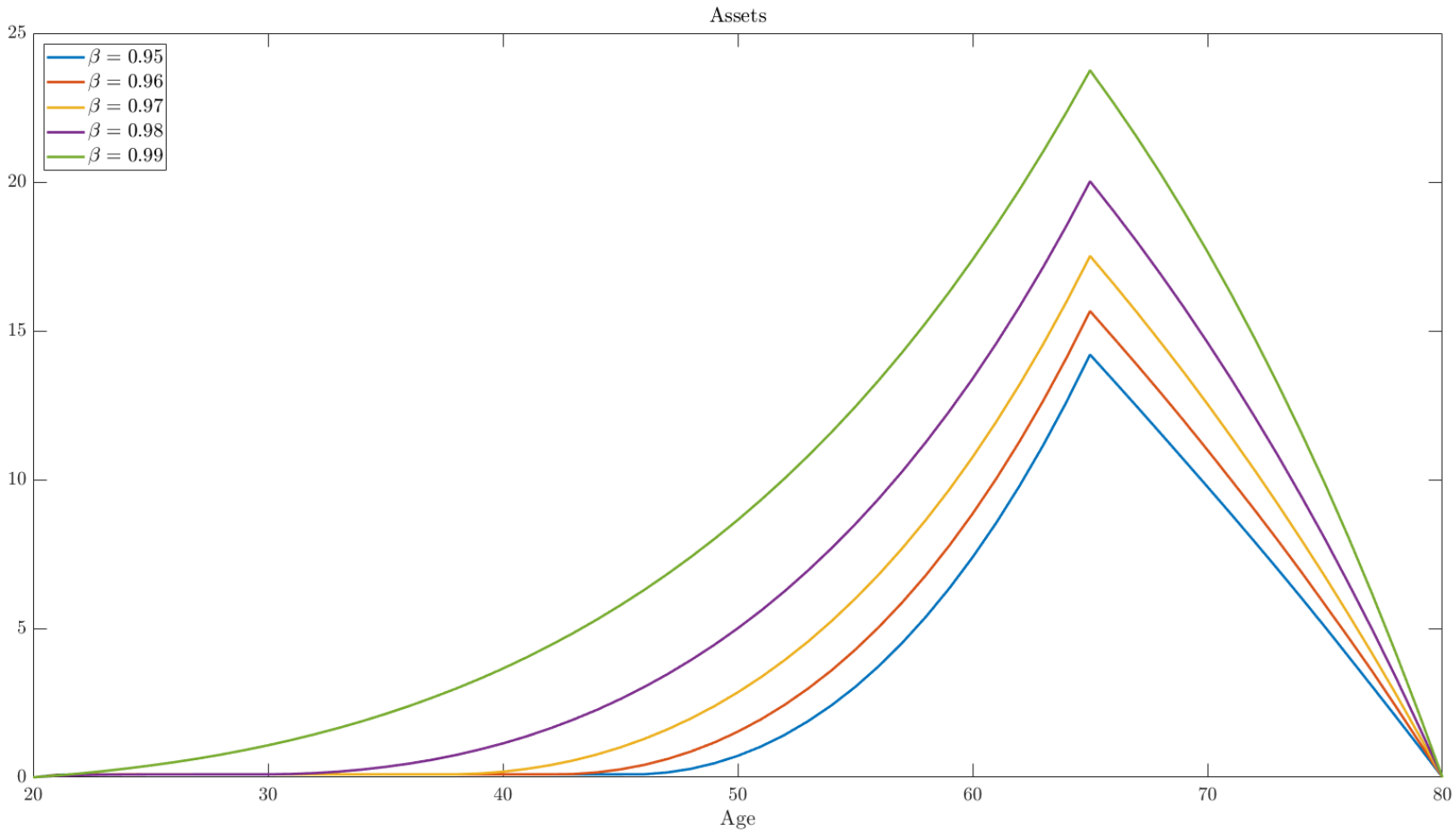
Ruling out Negative Net Assets

- What happens if banks are unwilling to let people have negative assets?
- But banks are generally reluctant to let people have negative net assets based on a promise that future labour income will pay off the loans.
- The charts on the following pages describe what happens when we impose the constraint that assets have to be positive.
- Note that if we interpret assets as really meaning net wealth (assets minus liabilities) then we are not ruling out borrowing. People who buy a house with a 100 percent mortgage have unchanged levels of net assets because the asset they have acquired (the house) offsets the loan they have taken on.
- The most patient households (those with $\beta = 0.99$ or above) are not affected by this constraint because they never planned to borrow anyway.
- But everyone else goes through a period when they are "liquidity constrained". They would like to borrow to finance higher consumption but they can't. They choose to consume all their labour income and don't accumulate assets.
- The most impatient households in this model are liquidity constrained until their mid-40s and only start saving for retirement then.

Life Cycle Model: Consumption Paths with Negative Net Assets Not Allowed ($r = 1/0.97 - 1$)



Life Cycle Model: Asset Paths with Negative Net Assets Not Allowed ($r = 1/0.97 - 1$)



Implications for Fiscal Policy

- This model has strong implications for the effects of fiscal stimulus via tax cuts or rebates or "stimulus cheques".
- The model predicts that some people—those who are young and particularly the impatient young—will spend all of their stimulus.
- But everyone else will treat it more like "permanent income" consumers, only consuming that part of it that raises their long-run income.
- So the response to fiscal stimulus is likely to be larger than it would be if everyone was a permanent income consumer.
- It is also likely to depend on factors such as demographics and the willingness of the financial system to allow people to carry debt.

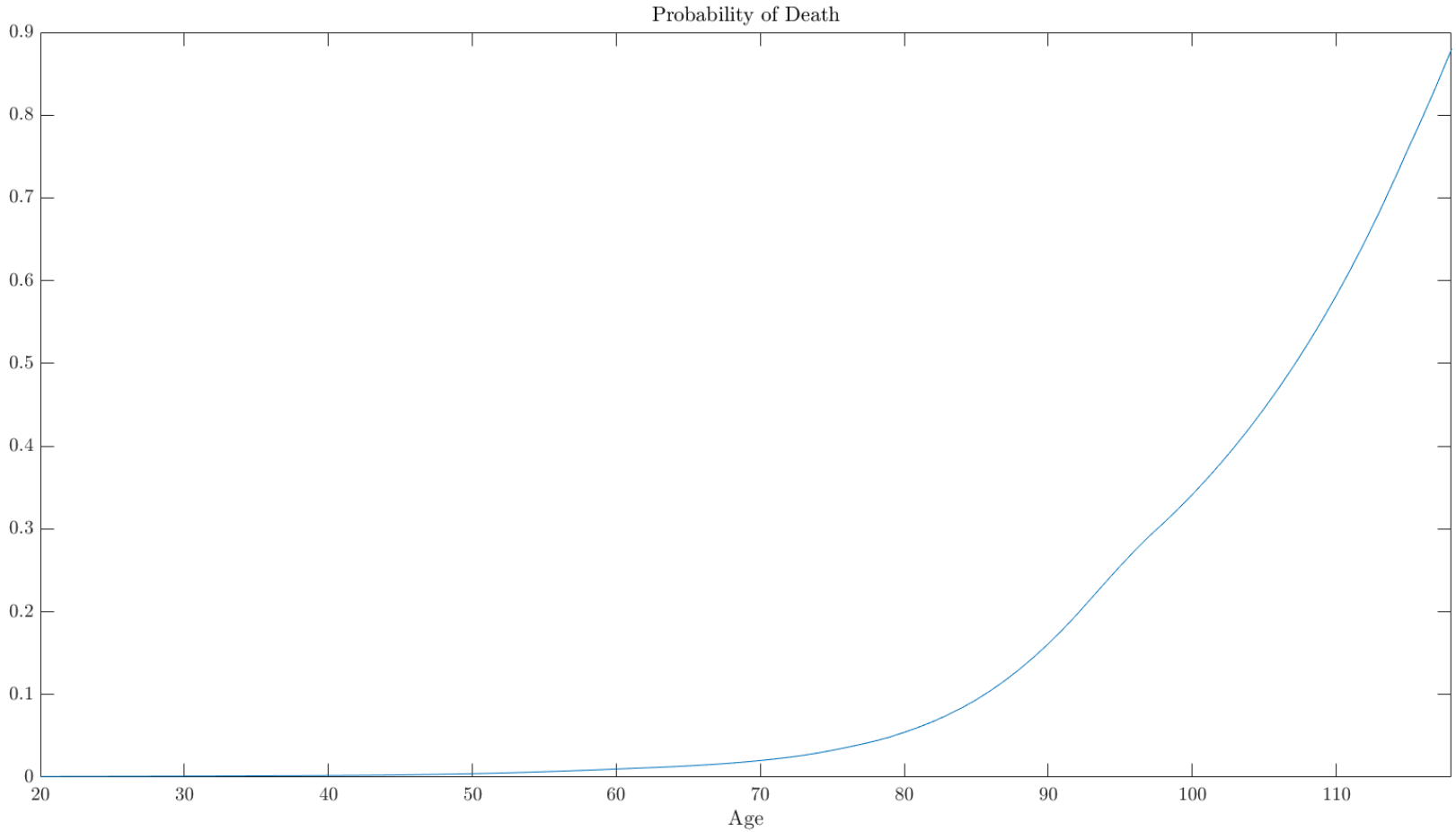
Uncertain Time of Death

- It is easy to adapt our model to allow uncertainty about the time of death.
- We can adapt our Bellman equation to take the form

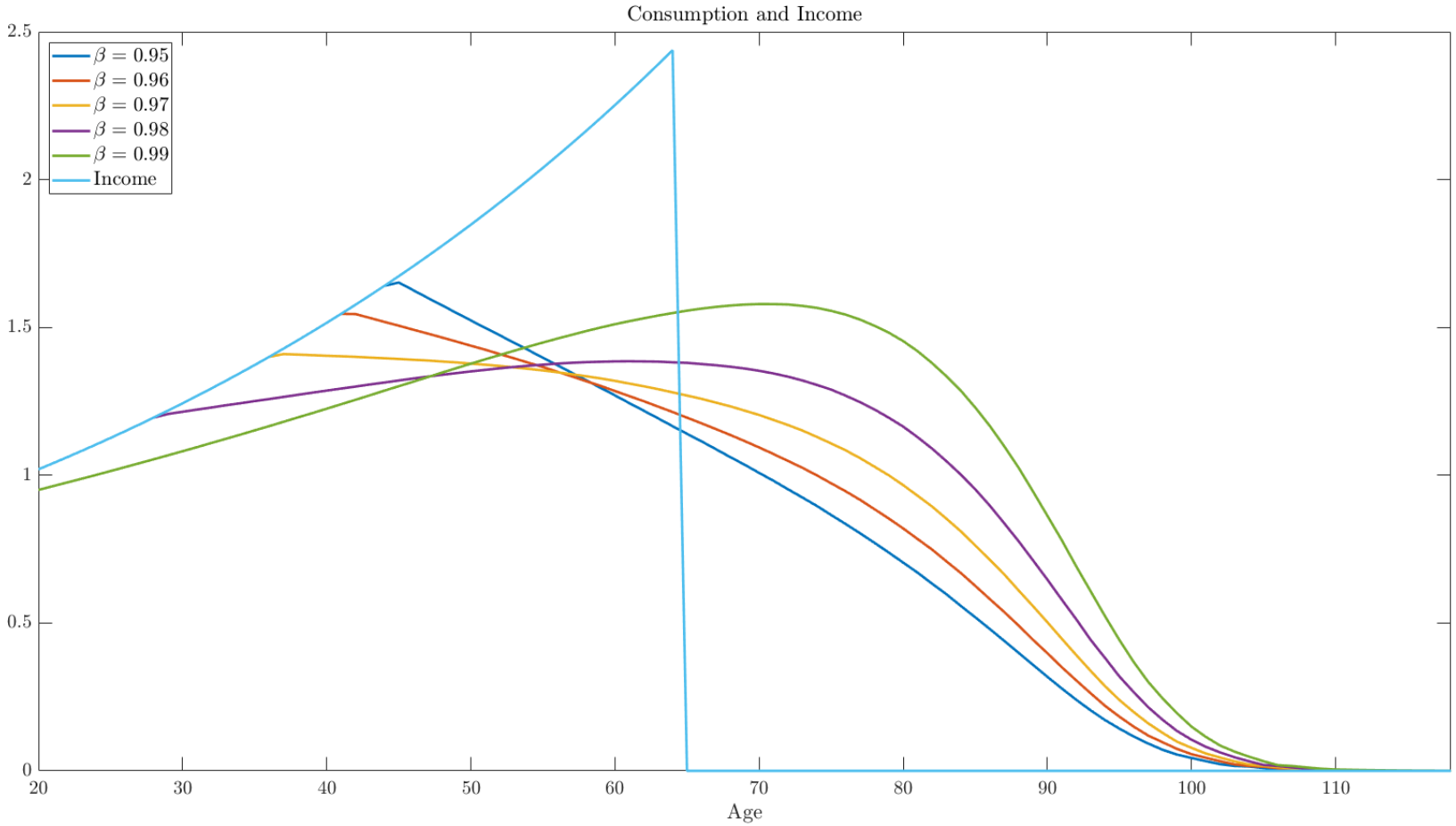
$$V_t(A_t) = \overset{MAX}{\{C_t\}} \left[\frac{C_t^{1-\gamma}}{1-\gamma} + \beta(1-\delta_t) V_{t+1}(A_{t+1}) \right]$$

  where δ_t is an age-varying probability of death, measuring the probability of people who reach age t but not age $t+1$. So people consider next period but place a lower weight on it because there is a chance it never happens.
- I have adapted our programme to incorporate 2019 data on probabilities of death from the US Social Security Administration. I have imposed a maximum feasible age of 120.
- The next few charts show the probability of death that has been used and repeats the life-cycle paths for consumption and assets, now understood as the consumption and assets of those who live to the various ages.
- Relative to our model with a fixed life length of 80 years, consumption earlier in life is lower and saving from income starts earlier because of the possibility that people will need assets to sustain them beyond 80 years of age.

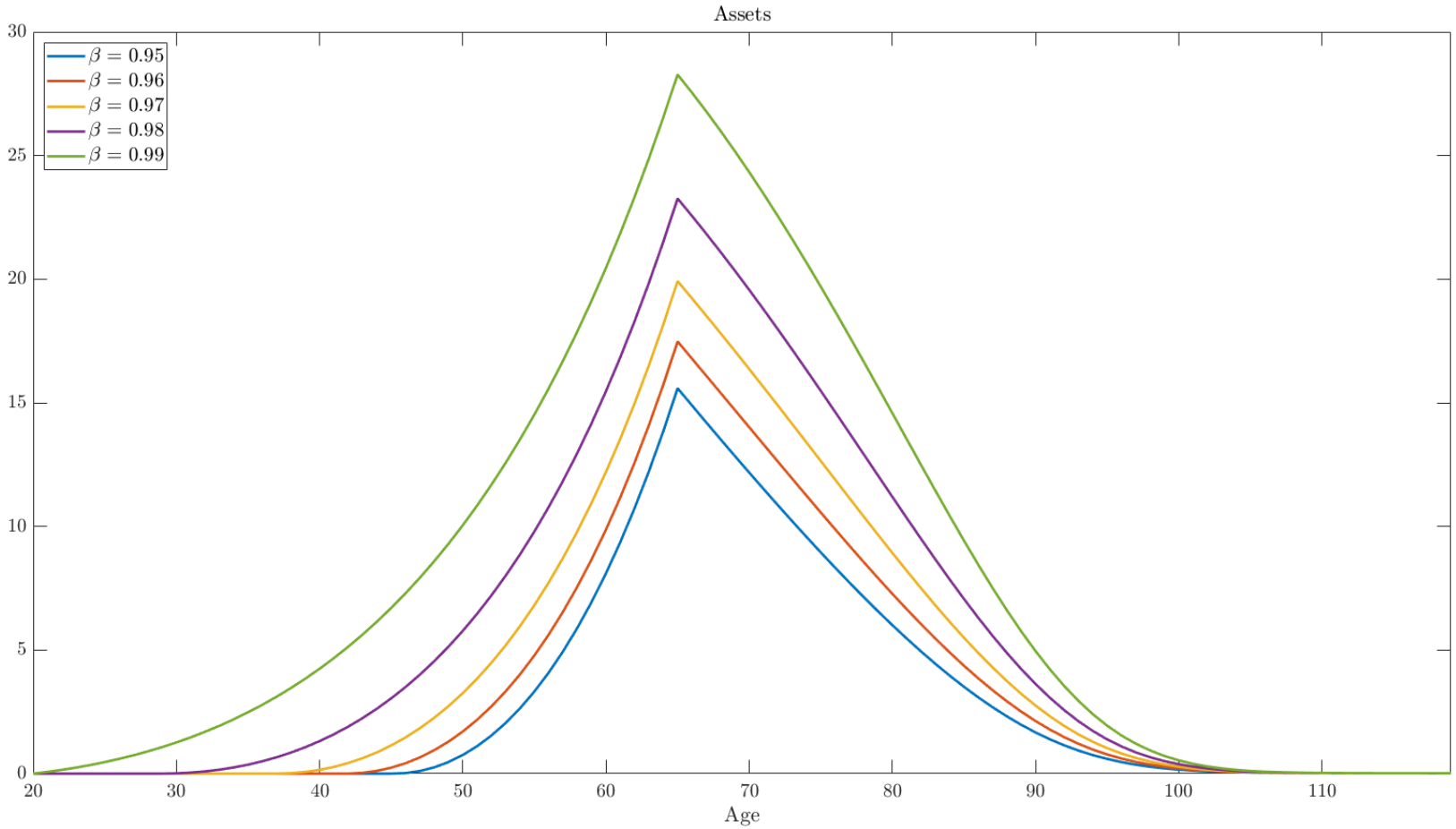
2019 US Data on Probability of Death by Age



Life Cycle Model: Consumption Paths with Uncertain Time of Death ($r = 1/0.97 - 1$)



Life Cycle Model: Asset Paths with Uncertain Time of Death ($r = 1/0.97 - 1$)



Bequests

- The final chart shows the amount of assets accumulated at each age by those who have died that period.
- These will presumably be left as bequests.
- Note that the model generates large amounts of bequests even though we haven't given our agents a bequest motive, i.e. they get utility from knowing they are leaving money to heirs.
- This matches data showing bequests being a major source of wealth acquisition.
- Another adaptation we could make to the model is to add uncertain out-of-pocket health expenses that rise in probability the longer you live.
- De Nardi, French and Jones's paper "Why Do the Elderly Save? The Role of Medical Expenses" (2010, Journal of Political Economy) uses dynamic programming to provide realistic modelling of mortality rates and the potential costs of health expenses and explore the role of the Medicaid system in impacting asset accumulation.

Life Cycle Model: Bequests by Age ($r = 1/0.97 - 1$)