Computational Economics with Python

Columbia University March 2018

John Stachurski

Lecture 3

Plan

Applications.

1. Optimal savings / income fluctuation problem
2. Job search / optimal stopping
3. Asset pricing

Using JIT and parallelization (and well chosen algorithms)

Application I: Optimal Savings (Plain Vanilla)

A household chooses $\{c_t\}_{t \geqslant 0}$ to maximize

$$\mathbb{E} \sum_{t=0}^{\infty} \beta^t u(c_t)$$

$$c_t + x_{t+1} \leqslant R x_t + w z_t,$$

with $c_t \geqslant 0$, $x_t \geqslant 0$ for all t

- $\beta \in (0, 1)$ is the discount factor
- x_t is asset holdings at t and c_t is consumption
- $w z_t$ is wages
- $R := 1 + r$ with $r > 0$

The exogenous state $\{z_t\}$ obeys

$$\ln z_{t+1} = \rho \ln z_t + d + \sigma \eta_{t+1} \quad \text{with} \quad \{\eta_t\} \stackrel{\text{IID}}{\sim} N(0,1)$$

We'll discretize to produce finite state Markov chain, with

$$Q(z, z') = \mathbb{P}\{z_{t+1} = z' \mid z_t = z\}$$

The utility function will be $u(c) = c^{1-\gamma}/(1-\gamma)$

The value function v^* is defined by $v^*(x, z) := \sup \sum_{t \geqslant 0} \beta^t \mathbb{E}\, u(c_t)$

The sup is over all feasible paths from (x, z)

By Bellman's principle of optimality, the value function satisfies

$$v^*(x,z) = \max_{0 \leqslant x' \leqslant Rx+zw} \left\{ u(Rx + wz - x') + \beta \sum_{z'} v^*(x',z') Q(z,z') \right\}$$

We discretize the state space to finite set $0 < x_1 < \cdots < x_n$

The Bellman operator is

$$Tv(x,z) = \max_{0 \leqslant x' \leqslant Rx+zw} \left\{ u(Rx + wz - x') + \beta \sum_{z'} v(x',z') Q(z,z') \right\}$$

Computing the value function via VFI:

- See `day3/optimal_saving`

Notes

- Avoid vectorization!
- Run this on a machine with lots of cores and watch them light up :-)

Application II: Job Search

We study a model of job search with persistent and transitory components to wages

Wages are given by

$$W_t = \exp(Z_t) + Y_t$$

where

- $Y_t \sim \exp(\mu + s\zeta_t)$
- $Z_{t+1} = d + \rho Z_t + \sigma \epsilon_{t+1}$
- ζ_t and ϵ_t are both IID and $N(0,1)$

The value function is

$$v(w,z) = \max\left\{\frac{u(w)}{1-\beta},\ u(c) + \beta\,\mathbb{E}_z\, v(w',z')\right\}$$

Standard VFI approach

- solve for value function by VFI
- compute optimal policy

But we can cut down the state size by half

$\implies$ 1–2 **orders of magnitude speed gain**!

The **continuation value function** is

$$f(z) := u(c) + \beta \, \mathbb{E}_z \, v(w', z')$$

By Bellman's equation, the value function satisfies

$$v(w, z) = \max \left\{ \frac{u(w)}{1 - \beta}, \; f(z) \right\}$$

Hence

$$f(z) = u(c) + \beta \, \mathbb{E}_z \max \left\{ \frac{u(w')}{1 - \beta}, f(z') \right\}$$

A functional equation in f — and a contraction

- In only one dimension!

Given f, can solve by stopping when

$$\frac{u(w)}{1-\beta} \geqslant f(z)$$

For utility we take $\mathrm{u(c)} = \mathrm{ln(c)}$

The reservation wage is the wage where equality holds, or

$$w^*(z) = \exp(f^*(z)(1-\beta))$$

Our aim is to solve for the reservation rule

Implementation

This time we avoid discretization

Use

- linear interpolation for function approximation
- Monte Carlo for integration

(Not necessary here but good when size of state $\geqslant 2$)

- See `day3/job_search`

Application III: Asset Pricing

The time t price of a claim to payoff G_{t+1} is typically expressed as

$$P_t = \mathbb{E}_t \left[M_{t+1} G_{t+1} \right]$$

where M_{t+1} is called the **stochastic discount factor**, or SDF

The special case $\beta = M_{t+1}$ is the risk neutral case

The another important special case is

$$M_{t+1} = \beta \frac{u'(C_{t+1})}{u'(C_t)}$$

This is the SDF derived in Lucas (1978)

Pricing Dividend Streams with Risk Aversion

Now let's price a claim to the dividend stream $\{D_t\}$

Ex-dividend: a claim to D_{t+1} and right to sell next period

Hence $G_{t+1} = D_{t+1} + P_{t+1}$

The price now satisfies

$$P_t = \mathbb{E}_t \left[M_{t+1}(D_{t+1} + P_{t+1}) \right]$$

Our aim is to solve for $\{P_t\}$ given $\{M_t, D_t\}$

To solve

$$P_t = \mathbb{E}_t[M_{t+1}(D_{t+1} + P_{t+1})]$$

let's assume that

- $D_{t+1} = d(X_{t+1})$ for some nonnegative function d
- $M_{t+1} = m(X_t, X_{t+1})$ for some positive function m
- $\{X_t\}$ is a finite Markov chain with stochastic matrix Q

Guessing a solution of the form $P_t = p(X_t)$, we aim to solve

$$p(X_t) = \mathbb{E}\left[m(X_t, X_{t+1})(d(X_{t+1}) + p(X_{t+1})) \mid X_t\right]$$

Suffices to find a p satisfying

$$p(x) = \sum_{y \in \mathbb{X}} m(x,y)\,[d(y) + p(y)]\,Q(x,y)$$

for all $x \in \mathbb{X}$

Equivalently, with

$$K(x,y) := m(x,y) Q(x,y)$$

we seek a p that solves

$$p(x) = \sum_{y \in \mathbb{X}} [d(y) + p(y)]\,K(x,y)$$

for all $x \in \mathbb{X}$

Treating $K(x, y)$ as a matrix, we can stack the equations

$$p(x) = \sum_{y \in \mathbb{X}} [d(y) + p(y)] \, K(x, y)$$

to obtain

$$p = Kd + Kp$$

This equation has the unique solution

$$p = (I - K)^{-1} K d$$

whenever $r(K) < 1$

- See `day3/markov_asset_tauchen.ipynb`

Asset Pricing with Nonstationary Dividends

In reality dividends are typically nonstationary

A standard model is

$$\ln \frac{D_{t+1}}{D_t} = \kappa(X_t, \eta_{t+1})$$

where

- $\{X_t\}$ is a stationary Markov process
- $\{\eta_t\} \overset{\text{IID}}{\sim} \phi$

Now prices are nonstationary, so we solve instead for the price dividend ratio

Start with

$$P_t = \mathbb{E}_t \left[M_{t+1} (D_{t+1} + P_{t+1}) \right]$$

The price-dividend ratio is

$$\frac{P_t}{D_t} = \mathbb{E}_t \left[M_{t+1} \frac{D_{t+1}}{D_t} \left(1 + \frac{P_{t+1}}{D_{t+1}} \right) \right]$$

With $V_t := P_t / D_t$ we have

$$V_t = \mathbb{E}_t \left[M_{t+1} \exp(\kappa(X_t, \eta_{t+1})) \left(1 + V_{t+1} \right) \right]$$

In solving

$$V_t = \mathbb{E}_t \left[M_{t+1} \exp(\kappa(X_t, \eta_{t+1})) \left(1 + V_{t+1}\right) \right]$$

let's assume that

- $M_{t+1} = m(X_t, \eta_{t+1})$ for some positive function m
- $\{X_t\}$ is a finite Markov chain with stochastic matrix Q

It then suffices to find a function v such that

$$v(x) = \sum_{y \in \mathbb{X}} \int m(x, \eta) \exp(\kappa(x, \eta)) \phi(d\eta) \left[1 + v(y)\right] Q(x, y)$$

for all $x \in \mathbb{X}$

To repeat, we seek a v that solves

$$v(x) = \sum_{y \in \mathbb{X}} \int m(x, \eta) \exp(\kappa(x, \eta)) \phi(d\eta) \, [1 + v(y)] \, Q(x, y)$$

for all $x \in \mathbb{X}$

Equivalently, with

$$A(x, y) := Q(x, y) \int m(x, \eta) \exp(\kappa(x, \eta)) \phi(d\eta)$$

we seek a v that solves

$$v(x) = \sum_{y \in \mathbb{X}} [1 + v(y)] \, A(x, y)$$

Treating

- A as a matrix with i, j-th element $A(x_i, x_j)$ and
- v as a column vector with i-th element $v(x_i)$

this becomes

$$v = A\mathbb{1} + Av$$

Here $\mathbb{1}$ is an $n \times 1$ column vector of ones

This equation has the unique solution

$$v = (I - A)^{-1} A\mathbb{1}$$

whenever $r(A) < 1$

Example. Consider the dividend process

$$\ln \frac{D_{t+1}}{D_t} = \kappa(X_t, \eta_{t+1}) = \mu_d + X_t + \sigma_d \eta_{d,t+1}$$

Here $\{\eta_{d,t}\} \overset{\text{IID}}{\sim} N(0,1)$

The state process $\{X_t\}$ obeys

$$X_{t+1} = \rho X_t + \sigma \xi_{t+1}$$

where $\{\xi_t\}$ is IID and standard normal

We can discretize it using **Tauchen's method**

Consumption is also nonstationary, obeying

$$\ln \frac{C_{t+1}}{C_t} = \mu_c + X_t + \sigma_c \eta_{c,t+1} \quad \text{where} \quad \{\eta_{c,t}\} \overset{\text{IID}}{\sim} N(0,1)$$

We use the Lucas SDF

$$M_{t+1} = \beta \frac{u'(C_{t+1})}{u'(C_t)}$$

The utility function is $u(c) = c^{1-\gamma}/(1-\gamma)$

Hence

$$M_{t+1} = \beta \left(\frac{C_{t+1}}{C_t} \right)^{-\gamma} = \beta \exp(-\gamma(\mu_c + X_t + \sigma_c \eta_{c,t+1}))$$

Recall the defintion

$$A(x,y) := Q(x,y) \int m(x,\eta) \exp(\kappa(x,\eta)) \phi(d\eta)$$

In our case this is

$$A(x,y) = \beta \exp\left(-\gamma\mu_c + \mu_d + (1-\gamma)x + \frac{\gamma^2\sigma_c^2 + \sigma_d^2}{2} \right) Q(x,y)$$

Now check $r(A) < 1$ and solve via $v = (I - A)^{-1} A \mathbb{1}$

- See `day3/asset_nonstationary_discretized.ipynb`

Large State Spaces

Recall: the price-dividend ratio is a v that solves

$$v(x) = \sum_{y \in \mathbb{X}} \int m(x, \eta) \exp(\kappa(x, \eta)) \phi(d\eta) \, [1 + v(y)] \, Q(x, y)$$

for all $x \in \mathbb{X}$

Equivalently, with

$$A(x, y) := Q(x, y) \int m(x, \eta) \exp(\kappa(x, \eta)) \phi(d\eta)$$

we seek a v that solves

$$v = (I - A)^{-1} A \mathbb{1}$$

But what if the state process has more dimensions, as in, say **Schorfheide, Song and Yaron, ECMA, 2018**?

$$\ln(C_{t+1}/C_t) = \mu_c + z_t + \sigma_{c,t}\,\eta_{c,t+1},$$

$$\ln(D_{t+1}/D_t) = \mu_d + \alpha z_t + \delta\sigma_{c,t}\,\eta_{c,t+1} + \sigma_{d,t}\,\eta_{d,t+1}$$

where

$$z_{t+1} = \rho\, z_t + (1-\rho^2)^{1/2}\,\sigma_{z,t}\,v_{t+1},$$

$$\sigma_{i,t} = \varphi_i\,\bar{\sigma}\exp(h_{i,t}),$$

$$h_{i,t+1} = \rho_{h_i} h_i + \sigma_{h_i}\xi_{i,t+1}, \quad i \in \{z, c, d\}$$

The state can be represented as the four dimensional vector

$$X_t := (z_t, h_{z,t}, h_{c,t}, h_{d,t})$$

Suppose that we discretize as follows:

$$z \to z^1, \dots, z^k, \qquad h_z \to h_z^1, \dots, h_z^k, \text{ etc.}$$

That means X_t can take k^4 different values

If $k = 25$, then $A = A(x, y)$ **is** $25^4 \times 25^4$

If A is $25^4 \times 25^4$, then it contains 25^8 floating point numbers

Each requires 8 bytes, so total memory consumption is

$$8 \times 25^8 = 1220703125000 = 1.2 \text{ terabytes}$$

Inverting it requires in the order of $25^{12} = 59604644775390625$ floating point opertions

6622 hours at 2.5 GHz

If we add another state variable then it becomes 103480286 hours $= 11812$ years ...

This is the **curse of dimensionality**

A Simulation-Based Approach

Recall that we are aiming to solve for the price-dividend ratio

$$V_t = \mathbb{E}_t \left[M_{t+1} \frac{D_{t+1}}{D_t} (1 + V_{t+1}) \right]$$

With $A_{t+1} = M_{t+1} \frac{D_{t+1}}{D_t}$,

$$V_t = \mathbb{E}_t \left[A_{t+1} (V_{t+1} + 1) \right]$$

Let's think about solving this using simulation

First rewrite our eq as

$$V_t = \mathbb{E}_t A_{t+1} + \mathbb{E}_t A_{t+1} V_{t+1}$$

Substitution gives

$$\begin{aligned} V_t &= \mathbb{E}_t A_{t+1} + \mathbb{E}_t A_{t+1} (\mathbb{E}_{t+1} A_{t+2} + \mathbb{E}_{t+1} A_{t+2} V_{t+2}) \\ &= \mathbb{E}_t A_{t+1} + \mathbb{E}_t A_{t+1} A_{t+2} + \mathbb{E}_t A_{t+1} A_{t+2} V_{t+2} \end{aligned}$$

Substituting again gives

$$\begin{aligned} V_t = \mathbb{E}_t A_{t+1} + \mathbb{E}_t A_{t+1} A_{t+2} + \mathbb{E}_t A_{t+1} A_{t+2} A_{t+3} \\ + \mathbb{E}_t A_{t+1} A_{t+2} A_{t+3} V_{t+3} \end{aligned}$$

The limit is

$$V_t = \mathbb{E}_t A_{t+1} + \mathbb{E}_t A_{t+1} A_{t+2} + \mathbb{E}_t A_{t+1} A_{t+2} A_{t+3} + \mathbb{E}_t A_{t+1} A_{t+2} A_{t+3} A_{t+4} + \cdots$$

Consolidating, the **forward solution** is

$$V_t^* = \mathbb{E}_t \left[\sum_{n=1}^{\infty} \prod_{i=1}^{n} A_{t+i} \right]$$

Exists if

$$\limsup_{n \to \infty} \mathbb{E}_t \left[\prod_{i=1}^{n} A_{t+i} \right]^{1/n} < 1$$

Note that

$$V_t^* = \mathbb{E}_t \left[\sum_{n=1}^{\infty} \prod_{i=1}^{n} A_{t+i} \right] = \mathbb{E}_{X_t} \left[\sum_{n=1}^{\infty} \prod_{i=1}^{n} A_{t+i} \right]$$

Written state by state, this becomes

$$v(x) = \mathbb{E}_x \left[\sum_{n=1}^{\infty} \prod_{i=1}^{n} A_i \right].$$

How can we calculate the right hand side?

Our proposal to calculate

$$v(x) = \mathbb{E}_x \left[\sum_{n=1}^{\infty} \prod_{i=1}^{n} A_i \right]$$

1. Fix large integers N and M
2. Generate M independent paths

$$A_1^{(m)}, \ldots A_N^{(m)},$$

and estimate $v(x)$ via

$$v_M(x) := \frac{1}{M} \sum_{m=1}^{M} \Lambda(x, N, m) \quad \text{where} \quad \Lambda(x, N, m) := \sum_{n=1}^{N} \prod_{i=1}^{n} A_i^{(m)}$$

Disadvantages of this method:

- slow relative to discretization **if** the state space is small

Advantages of this method:

- works in high dimensions
- “lazy” evaluation
- highly parallelizable

- See `day3/asset_pricing_simulation.ipynb`