Projection Methods & Iteration Schemes

Emile Alexandre Marin

emarin@ucdavis.edu

October 17, 2022

Re-re-revisiting the Neoclassical growth model

Consider the same model framework as last week

$$
\begin{aligned}
c_t^{-v} =& \mathrm{E}_t \left[\beta c_{t+1}^{-v} \alpha z_{t+1} k_{t+1}^{\alpha-1} \right] \\
c_t + k_{t+1} =& z_t k_t^{\alpha} \\
\ln\left(z_{t+1}\right) =& \rho \ln\left(z_t\right) + \varepsilon_{t+1} \\
& \varepsilon_{t+1} \sim N\left(0, \sigma^2\right) \\
& k_1, z_1 \text{ given}
\end{aligned}
$$

Re-re-revisiting the Neoclassical growth model

Consider the same model framework as last week

$$
\begin{aligned}
c_t^{-v} =& \mathrm{E}_t \left[\beta c_{t+1}^{-v} \alpha z_{t+1} k_{t+1}^{\alpha - 1} \right] \\
c_t + k_{t+1} =& z_t k_t^{\alpha} \\
\ln\left(z_{t+1}\right) =& \rho \ln\left(z_t\right) + \varepsilon_{t+1} \\
& \varepsilon_{t+1} \sim N\left(0, \sigma^2\right) \\
& k_1, z_1 \text{ given}
\end{aligned}
$$

- But this time, we utilize a different solution method

True rational expectations solution

- The policy functions are given by:

$$
\begin{aligned}
c_t &= c\left(k_t, z_t\right) \\
k_{t+1} &= k\left(k_t, z_t\right)
\end{aligned}
$$

True rational expectations solution

- The policy functions are given by:

$$c_t = c\left(k_t, z_t\right)$$
$$k_{t+1} = k\left(k_t, z_t\right)$$

- Using the Euler equation (optimality):

$$e\left(k_t, z_t\right) = -c_t^{-v} + \mathrm{E}_t\left[\beta c_{t+1}^{-v} \alpha z_{t+1} k_{t+1}^{\alpha-1}\right]$$

True rational expectations solution

- The policy functions are given by:

$$c_t = c(k_t, z_t)$$
$$k_{t+1} = k(k_t, z_t)$$

- Using the Euler equation (optimality):

$$e(k_t, z_t) = -c_t^{-v} + \mathrm{E}_t\left[\beta c_{t+1}^{-v} \alpha z_{t+1} k_{t+1}^{\alpha-1}\right]$$

$*$ Only at the correct solution $c(k_t, z_t)$ and $k(k_t, z_t)$:

$$e(k_t, z_t) = 0 \forall k_t, z_t$$

Logic of projection methods

$$c_t = c(k_t, z_t) \approx P_n(k_t, z_t; \eta_n)$$

- goal: solve for $P_n(k_t, z_t; \eta_n) \approx c(k_t, z_t)$

Logic of projection methods

$$c_t = c\left(k_t, z_t\right) \approx P_n\left(k_t, z_t; \eta_n\right)$$

- goal: solve for $P_n\left(k_t, z_t; \eta_n\right) \approx c\left(k_t, z_t\right)$
  - $P_n(\cdot)$: from class of approximating functions (polynomials or splines)
  - n is fixed $\Longrightarrow$ solve for η_n, a finite-dimensional object
  - structural parameters $(\alpha, \beta, \rho, \sigma)$ have fixed numerical values (thus not included as arguments in policy function)

Logic of projection methods

$$c_t = c(k_t, z_t) \approx P_n(k_t, z_t; \eta_n)$$

- goal: solve for $P_n(k_t, z_t; \eta_n) \approx c(k_t, z_t)$
  - $P_n(\cdot)$: from class of approximating functions (polynomials or splines)
  - n is fixed $\Longrightarrow$ solve for η_n, a finite-dimensional object
  - structural parameters $(\alpha, \beta, \rho, \sigma)$ have fixed numerical values (thus not included as arguments in policy function)
- $k(k_t, z_t)$ implicitly defined by budget constraint

Logic of projection methods

$$c_t = c\left(k_t, z_t\right) \approx P_n\left(k_t, z_t; \eta_n\right)$$

- goal: solve for $P_n\left(k_t, z_t; \eta_n\right) \approx c\left(k_t, z_t\right)$
  - $P_n(\cdot)$: from class of approximating functions (polynomials or splines)
  - n is fixed $\Longrightarrow$ solve for η_n, a finite-dimensional object
  - structural parameters $(\alpha, \beta, \rho, \sigma)$ have fixed numerical values (thus not included as arguments in policy function)
- $k\left(k_t, z_t\right)$ implicitly defined by budget constraint
- One first-order equation left, namely Euler equation
  - NB: this is a different equation at each point in the state space $\Longrightarrow$ plenty of equations to determine $N+1$ coefficients

Peeling the Euler equation

- At M grid points $\{k_i, z_i\}$ with $M \geq N_n$ we would like the following to equal zero:

$$e\left(k_i, z_i; \eta_n\right) = -P_n\left(k_i, z_i; \eta_n\right)^{-v} +$$
$$\alpha\beta\times$$
$$E\left[\begin{array}{c} \mathrm{E} \\ P_n\left(\{\boldsymbol{k}'\}, \{\mathbf{z}'\}; \eta_n\right)^{-v} \times \\ \{\boldsymbol{z}'\} \times \\ \left(\{\boldsymbol{k}'\}\right)^{\alpha-1} \end{array}\right]$$

Peeling the Euler equation

- At M grid points $\{k_i, z_i\}$ with $M \geq N_n$ we would like the following to equal zero:

$$e\left(k_i, z_i; \eta_n\right) = -P_n\left(k_i, z_i; \eta_n\right)^{-v} +$$
$$\alpha\beta\times$$
$$E\left[\begin{array}{c} \mathrm{E} \\ P_n\left(\left\{\boldsymbol{k}'\right\}, \left\{\mathbf{z}'\right\}; \eta_n\right)^{-v} \times \\ \left\{\boldsymbol{z}'\right\} \times \\ \left(\left\{\boldsymbol{k}'\right\}\right)^{\alpha-1} \end{array}\right]$$

- Goal: $\forall$ grid points get an expression with η_n coefficients as only unknown

- Note that k_i and z_i are known

Expressing Euler as function $\{\eta_n\}$

$$
\mathrm{E}\left[\begin{array}{c}
e\left(k_i, z_i; \eta_n\right) = -P_n\left(k_i, z_i; \eta_n\right)^{-v} + \\
\alpha\beta\times \\
P_n\left(z_i k_i^{\alpha} - P_n\left(k_i, z_i; \eta_n\right), \exp\left\{\rho \ln\left(z_i\right) + \varepsilon'\right\}; \eta_n\right)^{-v} \times \\
\exp\left\{\rho \ln\left(z_i\right) + \varepsilon'\right\} \times \\
\left(z_i k_i^{\alpha} - P_n\left(k_i, z_i; \eta_n\right)\right)^{\alpha-1}
\end{array}\right]
$$

Expressing Euler as function $\{\eta_n\}$

$$\mathrm{E}\left[\begin{array}{c} e(k_i, z_i; \eta_n) = -P_n(k_i, z_i; \eta_n)^{-v} + \\ \alpha\beta\times \\ P_n(z_i k_i^{\alpha} - P_n(k_i, z_i; \eta_n), \exp\{\rho \ln(z_i) + \varepsilon'\}; \eta_n)^{-v} \times \\ \exp\{\rho \ln(z_i) + \varepsilon'\} \times \\ (z_i k_i^{\alpha} - P_n(k_i, z_i; \eta_n))^{\alpha-1} \end{array}\right]$$

- Still need to deal with expectations operator:

$$e(k_i, z_i; \eta_n) = -P_n(k_i, z_i; \eta_n)^{-v} +$$

$$\alpha\beta\times$$

$$P_n\left(z_i k_i^{\alpha} - P_n(k_i, z_i; \eta_n), \exp\left\{\rho \ln(z_i) + \sqrt{2}\sigma\zeta_j\right\}; \eta_n\right)^{-v} \times$$

$$\exp\left\{\rho \ln(z_i) + \sqrt{2}\sigma\zeta_j\right\} \times$$

$$(z_i k_i^{\alpha} - P_n(k_i, z_i; \eta_n))^{\alpha-1} (\omega_j/\sqrt{\pi})$$

where $\{\omega_j, \zeta_j\}_{j=1}^{J}$ are the Gauss-Hermite quadrature nodes

Finding coefficients $\{\eta_n\}$

Reduced infinite-dimensional problem (unknown function) to finding N scalar values

* $\{\eta_n\}$ chosen such that error terms are as small as possible.

Finding coefficients $\{\eta_n\}$

Reduced infinite-dimensional problem (unknown function) to finding N scalar values

* $\{\eta_n\}$ chosen such that error terms are as small as possible.

- True rational expect. solution gives zero error term $\forall\,(k_i, z_i)$
- Collocation $(M = N_n)$: Use equation solver to get errors exactly equal to zero on grid
- Galerkin $(M > N_n)$: Use minimization routine (and possibly smart weighting of error terms)

Solver vs. iIteration schemes

- In principle, can find coefficients by solving simultaneous equations (or minimization scheme).
- Instead, we will use iteration scheme (P_n appears on LHS and RHS of Euler):
    1. fixed-point iteration
    2. time iteration

Solver vs. ilteration schemes

- In principle, can find coefficients by solving simultaneous equations (or minimization scheme).
- Instead, we will use iteration scheme (P_n appears on LHS and RHS of Euler):
  1. fixed-point iteration
  2. time iteration

+ can deal with many coefficients
+ some iteration schemes are guaranteed to converge under reg. cond.
+ less of a black box

− does not use information on how best to update, e.g. like Newton Methods.

Iteration: Start with a grid

- Construct a grid with nodes for k and z
- At the nodes construct the basis functions of $P_n(k, z; \eta_n)$.
- For example, if

$$P_n(k, z; \eta_n) = \eta_{0,n} + \eta_{k,n}k + \eta_{z,n}z + \eta_{kk}k^2 + \eta_{kz}kz + \eta_{zz}z^2$$

then construct the matrix (where subscripts denote grid numbers)

$$X = \begin{bmatrix} 1 & k_1 & z_1 & k_1^2 & k_1 z_1 & z_1^2 \\ 1 & k_2 & z_2 & k_2^2 & k_2 z_2 & z_2^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & k_M & z_M & k_M^2 & k_M z_M & z_M^2 \end{bmatrix}$$

and calculate $(X'X)^{-1} X'$

Fixed Point Iteration Cookbook

The value of η_n used in the q^{th} iteration is referred to as η_n^q. At each grid point:

1. Calculate the RHS of the Euler equation using the latest value for η_n, i.e., η_n^{q-1}
2. Use RHS to calculate c_i, value for c at i^{th} grid point
3. Use values for c_i to obtain an estimate for η_n, denoted by $\hat{\eta}_n^q$
   - Polynomial: run a regression to get $\hat{\eta}_n^q$
   - Spline: the values of c at the nodes are the new values of η_n
4. Update coefficients slowly: $\eta_n^q = \lambda \hat{\eta}_n^q + (1-\lambda)\eta_n^{q-1}$

Updating coefficients (1/2)

- Calculate current consumption values implied by η_n^{j-1} at each grid point
- Use η_n^{q-1} to calculate $k' = z_i k_i^{\alpha} - P_n\left(k_i, z_i; \eta_n^{q-1}\right)$
- Use η_n^{q-1} to calculate $c' = P_n\left(k', z'; \eta_n^{q-1}\right)$
- Then, get c_i from:

$$(c_i)^{-v} =$$

$$P_n\left(z_i k_i^{\alpha} - P_n\left(k_i, z_i; \eta_n^{q-1}\right), \exp\left\{\rho \ln\left(z_i\right) + \sqrt{2}\sigma\zeta_j\right\}; \eta_n^{q-1}\right)^{-v} \times$$
$$\exp\left\{\rho \ln\left(z_i\right) + \sqrt{2}\sigma\zeta_j\right\} \times \left(z_i k_i^{\alpha} - P_n\left(k_i, z_i; \eta_n^{q-1}\right)\right)^{\alpha-1} \left(\omega_j/\sqrt{\pi}\right)$$

Updating coefficient (2/2)

Step 2: Get new estimate for η_n by running a projection step

- Let $Y = [c_1, c_2, \cdots, c_M]'$

$$P_n(k, z; \eta_n) = \eta_{0,n} + \eta_{k,n}k + \eta_{z,n}z + \eta_{kk}k^2 + \eta_{kz}kz + \eta_{zz}z^2$$

then

$$\hat{\eta}_n^q = (X'X)^{-1} X'Y$$

$$P_n(k, z; \eta_n) = \exp\left(\eta_{0,n} + \eta_{k,n}k + \eta_{z,n}z + \eta_{kk}k^2 + \eta_{kz}kz + \eta_{zz}z^2\right)$$

then

$$\widehat{\eta}_n^q = (X'X)^{-1} X' \underbrace{\ln}(Y)$$

$$\eta_n^q = \lambda\widehat{\eta}_n^q + (1-\lambda)\eta_n^{q-1} \text{ for } 0 < \lambda \leq 1$$

Convergence and Fixed Point Iteration

- Fixed-point iteration does not always converge
- Choosing a lower value of λ :
  - convergence more likely
  - slows down algorithm if lower value not needed for convergence

Convergence and Fixed Point Iteration

- Fixed-point iteration does not always converge
- Choosing a lower value of λ :
  - convergence more likely
  - slows down algorithm if lower value not needed for convergence

Alternative is time iteration

Time Iteration

- At each grid point use η_n^{q-1} only for next period's choices
- Again solve for c_i at each grid point
  - this is now a bit trickier (non-linear problem)
- Get n_n^q as with fixed-point iteration
- guaranteed to converge without dampening (under regularity conditions)

Time Iteration

- At each grid point use η_n^{q-1} only for next period's choices
- Again solve for c_i at each grid point
  - this is now a bit trickier (non-linear problem)
- Get n_n^q as with fixed-point iteration
- guaranteed to converge without dampening (under regularity conditions)

Solve c_i from following non-linear equation

$$(\boldsymbol{c_i})^{-v} = \sum_{j=1}^{J} \left[\begin{array}{c} \alpha\beta\times \\ P_n\left(z_i k_i^\alpha - c_i, \exp\left\{\rho \ln\left(z_i\right) + \sqrt{2}\sigma\zeta_j\right\}; \eta_n^{q-1}\right)^{-v} \times \\ \exp\left\{\rho \ln\left(z_i\right) + \sqrt{2}\sigma\zeta_j\right\} \times \\ \left(z_i k_i^\alpha - c_i\right)\right)^{\alpha-1} \\ \omega_j/\sqrt{\pi} \end{array} \right]$$

Fixed-point versus time iteration

- Fixed-point iteration uses η_n^{q-1} for all terms on the RHS, i.e. tomorrow's consumption choice and today's capital choice

Fixed-point versus time iteration

- Fixed-point iteration uses η_n^{q-1} for all terms on the RHS, i.e. tomorrow's consumption choice and today's capital choice
- Time iteration uses η_n^{q-1} only to evaluate next period's consumption

Fixed-point versus time iteration

- Fixed-point iteration uses η_n^{q-1} for all terms on the RHS, i.e. tomorrow's consumption choice and today's capital choice
- Time iteration uses η_n^{q-1} only to evaluate next period's consumption
- The structure of time iteration mimics the choice of value function iteration:
  - next period's behavior described by previous solution for value function
  - Bellman equation used to solve for choice of c and k simult.

Endogenous grid points

- Simple idea: construct grid for k' instead of a grid for k
- Instead of solving for the choice k' given k, we now solve for the value of k that would have led to the choice k'

Endogenous grid points

- Simple idea: construct grid for k' instead of a grid for k
- Instead of solving for the choice k' given k, we now solve for the value of k that would have led to the choice k'

In both cases you end up at each grid point with a set of values for k and a set of corresponding values for k'.

Endogenous grid points

- Simple idea: construct grid for k' instead of a grid for k
- Instead of solving for the choice k' given k, we now solve for the value of k that would have led to the choice k'

In both cases you end up at each grid point with a set of values for k and a set of corresponding values for k'.

- You can use endogenous grid points both with fixed-point and with time iteration
- The added value with time iteration lies in getting rid of the non-linear problem of solving for today's choices

Endogenous grid points and time iteration

- Time iteration $\implies$ use η_n^{q-1} for tomorrow's choices
- use η_n^q only for today's choices (which show up on both sides of the policy function

Then, get c_i from:

$$(c_i)^{-v} = \sum_{j=1}^{J} \left[\begin{array}{c} \alpha\beta\times \\ P_n\left(k_i', \exp\left\{\rho \ln(z_i) + \sqrt{2}\sigma\zeta_j\right\}; \eta_n^{q-1}\right)^{-v} \times \\ \exp\left\{\rho \ln(z_i) + \sqrt{2}\sigma\zeta_j\right\} \times \\ (k_i')^{\alpha-1}\left(\omega_j/\sqrt{\pi}\right) \end{array} \right]$$

and k_i from

$$k_i' + c_i = z_i k^\alpha$$

When can't you use projection methods?

- Not all solutions to optimization problems can be characterized by first-order conditions
- e.g. when objective function is not concave or budget set not convex

When can't you use projection methods?

- Not all solutions to optimization problems can be characterized by first-order conditions
- e.g. when objective function is not concave or budget set not convex

$\Longrightarrow$ use Value Function.

When can't you use projection methods?

- Not all solutions to optimization problems can be characterized by first-order conditions
- e.g. when objective function is not concave or budget set not convex

$\Longrightarrow$ use Value Function.

More challenges:

- Constructing a grid where all calculations are well defined may be tough
  - e.g., not get negative consumption/unemployment
- calculations should be possible also on path towards solution

$\Longrightarrow$ Endogenize grid using simulations (Parameterized expectations)

Where to from here?

- introduce idiosyncratic (agent specific) uncertainty (e.g. employment risk) – Aiyagari
  - * $\implies$ distribution of assets becomes object of interest
  - questions of wealth inequality
  - distribution is constant
- allow for aggregate uncertainty (e.g. productivity risk) – Krusell-Smith and others
  - * prices now depend on distribution $\implies$ individual policy function very high dimensional
  - Assume prices only depend on mean
  - Generate K' distribution using simulation of individual policy rules (Approximate aggregation)

References

- Heer, B., and A. Maussner, 2009, Dynamic General Equilibrium Modeling.
- Judd, K. L., 1998, Numerical Methods in Economics.
- Rendahl, P., Economic Journal 2015, Inequality constraints in recursive economies.
  - shows that time-iteration converges even in the presence of inequality constraints
- Wouter den Haan Notes *http://www.wouterdenhaan.com/notes*