

# Advanced Macroeconomics II

## Handout 6 - The Endogenous Grid Method

Sergio Ocampo

Western University

October 20, 2020

# Short recap

Prototypical DP problem:

$$\begin{aligned} V(z, k) &= \max_{\{c, k'\}} u(c) + \beta E \left[ V(z', k') \mid z \right] \\ \text{s.t. } c + k' &= f(z, k) \\ z' &= h(z, \eta); \eta \text{ stochastic} \end{aligned}$$

- ▶ We are looking for functions  $\mathbf{V}, \mathbf{g}^c, \mathbf{g}^k$ : We cannot solve this.

We need to solve an approximate problem:

- ▶ Approximate continuous function: **Interpolation**
  - ▶ Requires “exact” solution of maximization problem: **Optimization**
  - ▶ Requires computing expectations: **Integration**

# Why is VFI so costly?

$$V(z, k) = \max_{\{c, k'\}} u(c) + \beta E \left[ V(z', k') | z \right]$$

- ▶ Combination of maximization and expectations is lethal
- ▶ Optimizing requires \*a lot\* of function evaluations
  - ▶ Each function evaluation requires expectations, interpolations and often derivatives

## Key idea:

- ▶ Can we bypass the maximization step?
- ▶ Focus on the Euler equation

# Carroll (2006)

Maximization requires satisfying FOC:

$$u'(c) = \beta E \left[ V_k(z', k') | z \right] \quad c + k' = f(z, k)$$

Usual approach:

- ▶ Fix  $(z, k)$  and solve for  $(k', c)$
- ▶ Consumption is immediately given  $k'$ :  $c = f(z, k) - k'$
- ▶ Problem is to try a bunch of  $k'$  to solve

$$u'(f(z, k) - k') = \beta E \left[ V_k(z', k') | z \right]$$

# Carroll (2006)

Maximization requires satisfying FOC:

$$u'(c) = \beta E \left[ V_k(z', k') | z \right] \quad c + k' = f(z, k)$$

Carroll's approach:

- ▶ Fix  $(k', z)$  and solve for  $k$ ! Hence the endogenous grid name
- ▶ Problem is to solve:

$$f(z, k) = \underbrace{(u')^{-1} \left( \beta E \left[ V_k(z', k') | z \right] \right)}_{\text{Known given } (k', z)} + k'$$

- ▶ This is a nonlinear equation, but a simple one to solve

**Key:** Expectation and derivatives only taken once! No interpolations!

# Standard algorithm

---

## Algorithm 1: EGM: Standard Method

---

Function EGM( $V, \vec{k}, \vec{z}, parameters$ ):

```
for  $i=1:n_z$  do
  for  $j=1:n_k$  do
     $F(x) = f(\vec{z}_i, x) - \vec{k}_j - (u')^{-1} \left( \beta E \left[ V_k \left( z', \vec{k}_j \right) | \vec{z}_i \right] \right)$ 
    # Find [k_min, k_max], check corners, further bracket zero
     $k\_endo[j] = \text{Roots}(F, k\_min, k\_max)$ 
     $V\_endo[j] = u(f(\vec{z}_i, k\_endo[j]) - \vec{k}_j) + \beta E \left[ V \left( z', \vec{k}_j \right) | \vec{z}_i \right]$ 
  # Interpolate value function to exogenous grid
   $V\_new(i,:) = \text{Interpolation}(k\_endo, V\_endo, \vec{k})$ 
return  $V\_new$ 
```

---

# Change of variable - Know your states

- ▶ A change of variable makes things easier
- ▶ Define  $Y$  as total income, or cash on hand:  $Y = f(z, k)$
- ▶ We can change the state in our problem

$$\begin{aligned} V(z, Y) &= \max_{\{k'\}} u(Y - k') + \beta E[V(z', Y') | z] \\ \text{s.t. } Y' &= f(z', k') \\ z' &= h(z, \eta); \eta \text{ stochastic} \end{aligned}$$

- ▶ Control variable  $k'$  (partially) determines future state
- ▶ We still need to hang onto  $z$  as a state, why?

# Change of variable - Know your states

Note that  $Y'$  is a function of  $k'$  and  $z'$  so we can write

$$\mathbb{V}(z, k') = \beta E \left[ V(Y'(z', k'), z') | z \right]$$

$$\mathbb{V}_k(z, k') = \beta E \left[ V_Y(Y'(z', k'), z') \frac{\partial Y(z', k')}{\partial k} | z \right]$$

$$V(z, Y) = \max_{\{k'\}} u(Y - k') + \mathbb{V}(z, k')$$

$$\text{s.t. } Y' = f(z', k')$$

$$z' = h(z, \eta); \eta \text{ stochastic}$$



# Modified EGM

---

## Algorithm 2: EGM: Change of State Method

---

**Function**  $\text{EGM}(V, \vec{k}, \vec{z}, \text{parameters})$ :

# Note: You already know  $Y()$  for any  $()$ , let  $Y_{ij} = Y(\vec{z}_i, \vec{k}_j)$

**for**  $i=1:n_z$  **do**

**for**  $j=1:n_k$  **do**

$\mathbb{V}_j = \beta E \left[ V \left( Y(z', \vec{k}_j), z' \right) \mid \vec{z}_i \right]$

$\vec{c}_{\text{endo}} = (u')^{-1} \cdot (\mathbb{V}_k)$  (Note: Evaluating whole vector)

$\vec{Y}_{\text{endo}} = \vec{c}_{\text{endo}} + \vec{k}$

$\vec{V}_{\text{endo}} = u(\vec{c}_{\text{endo}}) + \mathbb{V}$

    # Interpolate value function to exogenous grid

$V_{\text{new}}[i,:] = \text{Interpolation}(\vec{Y}_{\text{endo}}, \vec{V}_{\text{endo}}, \vec{Y}[i,:])$

Change variable back to  $k$ . Note:  $V_{ji} = V(Y(\vec{z}_i, \vec{k}_j), z_i) = V(\vec{z}_i, \vec{k}_j)$

---

# Some comments

The method still has some flexibility

1. How to compute derivatives (you can get it from interpolation step)
2. How to compute expectations (no interpolation if  $z$  is discrete)
3. How to judge convergence (standard practice is actually to pass  $\mathbb{V}$  along and judge convergence with it)
4. How to map to capital after convergence
  - 4.1 Interpolation of  $V(Y_{endo})$  to  $V(Y_{exo})$ :  $Y_{exo}$  maps to  $k$  by construction
  - 4.2 Keep  $Y_{endo}$ . Solve for  $\vec{k}(z)$  s.t.  $Y_{endo}(z) = f(z, \vec{k}(z))$

# Labor Supply

# Labor supply adds an equation

$$\begin{aligned} V(z, k) &= \max_{\{c, k'\}} u(c, \ell) + \beta E \left[ V(z', k') | z \right] \\ \text{s.t. } c + k' &= f(z, k, \ell) \\ z' &= h(z, \eta); \eta \text{ stochastic} \end{aligned}$$

FOC:

$$u_c(c, \ell) = \beta E \left[ V_k(z', k') | z \right] \quad -u_\ell(c, \ell) = f_\ell(z, k, \ell) \quad c + k' = f(z, k, \ell)$$

# Attempting EGM - Problems

Change of variable:

$$Y(z, k) = f(z, k, \ell(z, k)) = zk^\alpha \ell(z, k)^{1-\alpha} + (1 - \delta)k$$

- ▶ Cannot define exogenous grid for  $Y$ . Grid depends on policy function.

Euler equation:

$$u_c(c, \ell(z, k)) = \beta E \left[ V_k(z', k') | z \right]$$

- ▶ In general, cannot invert this equation for  $c$ .
  - ▶ Special case for additively separable preferences:  $u_c(c, \ell) = u_c(c)$

If only we knew  $\ell(z, k)$  we could almost use EGM!

# Barillas & Fernandez-Villaverde (2007)

**Idea:** Mix EGM and VFI in the spirit of Howard's policy function iteration

1. Fix a policy function for labor  $\ell_0(z, k)$  (a good guess is  $\ell_0(z, k) = \ell_{ss}$ )
2. Conduct  $N$  steps of EGM given  $\ell_0(z, k)$  (say  $N = 10$ )
  - ▶ EGM has to be modified to include labor.
  - ▶ We need to
3. Conduct  $M$  steps of VFI (say  $M = 1$ ) on the exogenous capital grid.
4. Replace  $\ell(z, k)$  with the output of step 3 and conduct step 2.

---

**Algorithm 3:** EGM: Fixed labor supply

---

**Function** EGM( $V, \vec{k}, \vec{z}, \ell(z, k), k'(z, k), parameters$ ):

**for**  $i=1:n_z$  **do**

**for**  $j=1:n_k$  **do**

1. Solve for  $\tilde{k}_{ij}$  s.t.  $k'(\tilde{k}_j) = \vec{k}_j$
2. Evaluate for labor from old policy:  $\tilde{\ell}_{ij} = \ell(\vec{z}_i, \tilde{k}_{ij})$
3. Evaluate expected value:  $\mathbb{V} = \beta E \left[ V(z', \vec{k}_j) \mid \vec{z}_i \right]$
4. Recover consumption (analytical solution):  $u_c(\tilde{c}_{ij}, \tilde{\ell}_{ij}) = \mathbb{V}_k$
5. Find endogenous capital  $\hat{k}_{ij}$  s.t.:  $\tilde{c}_{ij} + \vec{k}_j = f(\vec{z}_i, \hat{k}_{ij}, \tilde{\ell}_{ij})$
6. Update value at endogenous grid:  $V(\vec{z}_i, \hat{k}_{ij}) = u(\tilde{c}_{ij}, \tilde{\ell}_{ij}) + \mathbb{V}$

  7. Interpolate to exogenous grid:  $V\_new[i,:] = \text{Interp}(\hat{k}, V, \vec{k})$

---

## Some comments

1. In step 1 you can get the inverse from an interpolation routine
  - ▶ One option is to use a root finder
  - ▶ If you are using your own routine (say Cubic Splines) you can code your own inverse function
2. In step 5 we can be more ambitious and also update labor.
  - 5'. Find endogenous capital  $\hat{k}_{ij}$  s.t.:  $\tilde{c}_{ij} + \vec{k}_j = f(\vec{z}_i, \hat{\mathbf{k}}_{ij}, \ell(\vec{z}_i, \hat{\mathbf{k}}_{ij}))$ 
    - ▶ Doing this implies adding an interpolation step to the root finding
    - ▶ It also provides a better update of the value function
  - 6'. Update value at endogenous grid:  $V(\vec{z}_i, \hat{k}_{ij}) = u(\tilde{c}_{ij}, \ell(\vec{z}_i, \hat{\mathbf{k}}_{ij})) + \mathbb{V}$
3. Step 4 can be simplified if utility is separable  $u(c, \ell) = U(c) - H(\ell)$ 
  - ▶ In fact, all the algorithm gets easier
  - ▶ No need to carry around the policy function for labor



---

**Algorithm 4:** EGM: Endogenous labor supply with separable utility

---

**Function** EGM( $V, \vec{k}, \vec{z}, parameters$ ):

**for**  $i=1:n_z$  **do**

**for**  $j=1:n_k$  **do**

1. Evaluate expected value:  $\mathbb{V} = \beta E \left[ V \left( z', \vec{k}_j \right) \mid \vec{z}_i \right]$
2. Recover consumption (analytical solution):  $\tilde{c}_{ij} = (U')^{-1} (\mathbb{V}_k)$
3. Find  $(\hat{k}_{ij}, \tilde{\ell}_{ij})$  that solve FOC:
  - 3a. Define  $\hat{k}(\ell)$  analytically from FOC:  $\frac{-H'(\ell)}{U'(\tilde{c}_{ij})} = f_\ell(\vec{z}_i, \hat{\mathbf{k}}(\ell), \ell)$
  - 3b. Solve for  $\tilde{\ell}_{ij}$  numerically s.t.:  $\tilde{c}_{ij} + \vec{k}_j = f \left( \vec{z}_i, \hat{\mathbf{k}}(\tilde{\ell}_{ij}), \tilde{\ell}_{ij} \right)$
  - 3c. Assign endogenous grid point  $\hat{k}_{ij} = \hat{k}(\tilde{\ell}_{ij})$
4. Update value at endogenous grid:  $V(\vec{z}_i, \hat{k}_{ij}) = u \left( \tilde{c}_{ij}, \tilde{\ell}_{ij} \right) + \mathbb{V}$
5. Interpolate to exogenous grid:  $V\_new[i,:] = \text{Interp}(\hat{k}, V, \vec{k})$

# Envelope Condition Method

# Maliar & Maliar (2013)

First order conditions:

$$u_c(c, \ell) = \beta E \left[ V_k(z', k') | z \right] \quad (1)$$

$$\frac{-u_\ell(c, \ell)}{u_c(c, \ell)} = f_\ell(z, k, \ell) \quad (2)$$

$$c + k' = f(z, k, \ell) \quad (3)$$

Envelope condition:

$$V_k(k, a) = u_c(c, \ell) f_k(z, k, \ell) \quad (4)$$

Combining (1) and (4) we get a recursive equation for value derivative:

$$V_k(k, a) = \beta f_k(z, k, \ell) E \left[ V_k(z', k') | z \right] \quad (5)$$

**Key:** EGM works by solving (1), (2) and (3). ECM solves (4), (2) and (3). Equation (5) lets us update  $V_k$  directly without computing  $V$

# ECM - Algorithm - Inelastic labor supply

---

## Algorithm 5: ECM: Inelastic labor supply

---

Function ECM( $V_k, \vec{k}, \vec{z}, parameters$ ):

for  $i=1:n_z$  do

for  $j=1:n_k$  do

1. Get consumption analytically:  $c_{ij} = (u')^{-1} \left( \frac{V_k(\vec{z}_i, \vec{k}_j)}{f_k(\vec{z}_i, \vec{k}_j)} \right)$

Note: We are using present  $k$ , not future  $k$ .

2. Get  $k'$ :  $k'_{ij} = f(\vec{z}_i, \vec{k}_j) - c_{ij}$

3. Update  $V_k$ :  $V_k^{new}(\vec{z}_i, \vec{k}_j) = \beta f_k(\vec{z}_i, \vec{k}_j) E[V_k(z', k'_{ij}) | \vec{z}_i]$

Note: This step requires interpolation inside expectation

return  $V_k^{new}$ ;

---

# Some comments

1. No optimization or root finding in any step!
2. Careful when updating derivatives directly
  - ▶ You lose the power of the contraction mapping theorem
  - ▶ Particularly you lose uniqueness ( $V_k = 0$  is a fixed point)
  - ▶ Remember that your function is monotone, so  $V_k > 0$ !
3. Alternative is to update with  $V$  as we do always:
  - 0'. Get derivative of  $V$  at grid nodes  $V_k(\vec{z}_i, \vec{k}_j)$
  - 3'. Update value function:  $V^{new}(z, k) = u(c_{ij}) + \beta E \left[ V(z', k'_{ij}) | z \right]$ 
    - ▶ Note that you still need to do the interpolation inside the expectation
4. Authors say they get better results with  $V_k$ 
  - ▶ They reference another paper (Maliar & Maliar, 2012) that solves problems with 16 states using variants of the ECM

# ECM - Labor supply

---

**Algorithm 6:** ECM: Endogenous labor supply, Separable utility

---

**Function** ECM( $V_k, \vec{k}, \vec{z}, parameters$ ):

for  $i=1:n_z$  do

for  $j=1:n_k$  do

1. Get labor  $\ell_{ij}$  numerically:  $V_k(\vec{z}_i, \vec{k}_j) = \frac{H'(\ell_{ij})}{f_\ell(\vec{z}_i, \vec{k}_j, \ell_{ij})} f_k(\vec{z}_i, \vec{k}_j, \ell_{ij})$

Note: No interpolation or expectation

2. Get consumption analytically:  $c_{ij} = (U')^{-1} \left( \frac{H'(\ell_{ij})}{f_\ell(\vec{z}_i, \vec{k}_j, \ell_{ij})} \right)$

3. Get  $k'$ :  $k'_{ij} = f(\vec{z}_i, \vec{k}_j, \ell_{ij}) - c_{ij}$

3. Update  $V_k$ :  $V_k^{new}(\vec{z}_i, \vec{k}_j) = \beta f_k(\vec{z}_i, \vec{k}_j, \ell_{ij}) E[V_k(z', k'_{ij}) | \vec{z}_i]$

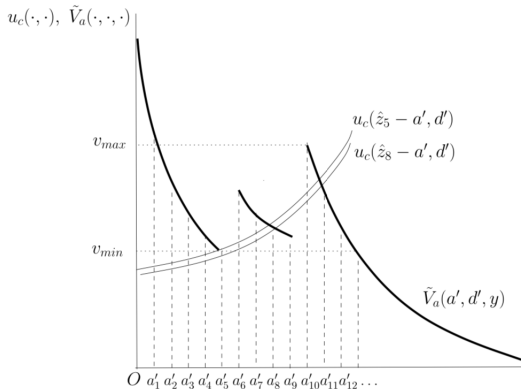
return  $V_k^{new}$ ;

---

# Extensions

# Non-Convex, Non-Smooth Problems - Fella (2014)

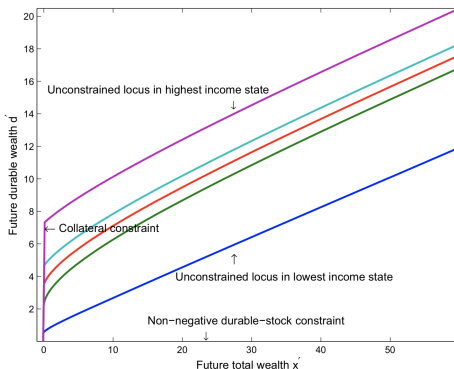
- ▶ Extend EGM to a problem with discrete state variable and continuous choices
- ▶ Discreteness is a problem because it generates kinks in the function
- ▶ Idea: EGM works away from the kinks!
- ▶ This is worth checking!





## 2 States+Borrowing Const. - Hintermaier & Koeniger (2010)

- ▶ Method for model with occasionally binding collateral constraints and non-separable utility in durable and non-durable consumption
- ▶ Good for applications with uninsurable income risk
- ▶ Idea: Solve the problem with a new state variable
  - ▶  $x$  : Cash on hand or beginning of period wealth



# Robustness

# Initial conditions are tricky

- ▶ EGM and ECM put a lot of trust on our guess of  $V$  and  $V_k$
- ▶ That trust is often misplaced during initial iterations
  - ▶ Initial guess won't capture curvature of the solution
- ▶ Always build a safety check
  - ▶ Your EGM or ECM might send you out of bounds
  - ▶ This often means negative savings
  - ▶ Under-estimate curvature at the bottom  $\rightarrow$  Over-estimate consumption
- ▶ If you are going out of bounds revert to traditional VFI