Modern Macro, Money, and International Finance

Eco529

Lecture 08: Numerical Methods

Andrey Alexandrov

Princeton University

Numerical Methods

1. Backward Equation: value function
    - General structure
    - Construction of the M matrix
    - VFI and PFI
2. Forward Equation: state variables
3. Jump processes and two-state problems

General Setup: Backward Equation

- Recall from previous lecture:

$$\rho^i v_t^i = \partial_t v_t^i + \left(\frac{c_t^i}{n_t^i} + (1-\gamma)\left(\Phi(\iota_t) - \delta - \gamma\frac{\sigma^2}{2} + \sigma\sigma_t^{v^i}\right)\right) v_t^i + \eta_t\mu_t^\eta\partial_\eta v_t^i + \frac{1}{2}(\eta_t\sigma_t^\eta)^2\partial_{\eta\eta} v_t^i$$

General Setup: Backward Equation

- Recall from previous lecture:

$$\rho^i v_t^i = \partial_t v_t^i + \left(\frac{c_t^i}{n_t^i} + (1-\gamma)\left(\Phi(\iota_t) - \delta - \gamma\frac{\sigma^2}{2} + \sigma\sigma_t^{v^i}\right)\right) v_t^i + \eta_t \mu_t^\eta \partial_\eta v_t^i + \frac{1}{2}(\eta_t \sigma_t^\eta)^2 \partial_{\eta\eta} v_t^i$$

- Generally interested in solving:

$$\rho v(\eta) = u(v,\eta) + \mu(v,\eta)\partial_\eta v(\eta) + \frac{1}{2}\sigma(v,\eta)^2 \partial_{\eta\eta} v(\eta)$$

- Discretized form:

$$\rho \mathbf{v} = \mathbf{u}(\mathbf{v}) + \mathbf{M}(\mathbf{v})\mathbf{v}$$

Constructing the M Matrix

- Grid $\{\eta_1, \dots, \eta_N\}$, $v = [v_1, \dots, v_N]'$, $\mu = [\mu_1, \dots, \mu_N]'$, etc
- Desired property: $M_{ii} \leq 0 \ \forall i$, $M_{ij} \geq 0 \ \forall i \neq j$
- Second derivative:
$$\partial_{\eta\eta} v_i = \frac{v_{i+1} - 2v_i + v_{i-1}}{(\Delta\eta)^2}$$
- First derivative: upwind scheme

$$\partial_\eta v_i^+ = \frac{v_{i+1} - v_i}{\Delta\eta}, \qquad \partial_\eta v_i^- = \frac{v_i - v_{i-1}}{\Delta\eta}$$
$$\partial_\eta v_i = v_i^+ \mathbf{1}_{\mu_i \geq 0} + v_i^- \mathbf{1}_{\mu_i < 0}$$

Constructing the M Matrix

- Denote $\mu_i^+ = \max\{\mu_i, 0\} \geq 0$, $\mu_i^- = \min\{\mu_i, 0\} \leq 0$

$$\rho v_i = u_i + \left[-\frac{\mu_i^-}{\Delta\eta} + \frac{\sigma_i^2}{2(\Delta\eta)^2}, \ \frac{\mu_i^- - \mu_i^+}{\Delta\eta} - \frac{\sigma_i^2}{(\Delta\eta)^2}, \ \frac{\mu_i^+}{\Delta\eta} + \frac{\sigma_i^2}{2(\Delta\eta)^2} \right] \begin{bmatrix} v_{i-1} \\ v_i \\ v_{i+1} \end{bmatrix}$$

- Any interior row of M sums to zero

Constructing the M Matrix

- Denote $\mu_i^+ = \max\{\mu_i, 0\} \geq 0$, $\mu_i^- = \min\{\mu_i, 0\} \leq 0$

$$\rho v_i = u_i + \left[-\frac{\mu_i^-}{\Delta\eta} + \frac{\sigma_i^2}{2(\Delta\eta)^2},\ \frac{\mu_i^- - \mu_i^+}{\Delta\eta} - \frac{\sigma_i^2}{(\Delta\eta)^2},\ \frac{\mu_i^+}{\Delta\eta} + \frac{\sigma_i^2}{2(\Delta\eta)^2} \right] \begin{bmatrix} v_{i-1} \\ v_i \\ v_{i+1} \end{bmatrix}$$

- Any interior row of M sums to zero
- At the boundaries:
  - Enforce $\sigma_1 = \sigma_N = 0$ and $\mu_1 \geq 0$, $\mu_N \leq 0$

Constructing the M Matrix

- Matrix M is by construction sparse $\Longrightarrow$ use sparse libraries:
  - Matlab: `spdiags`
  - Julia: `SparseArrays.spdiagm`
  - Python: `scipy.sparse.spdiags`

Constructing the M Matrix

- Matrix M is by construction sparse $\Longrightarrow$ use sparse libraries:
  - Matlab: `spdiags`
  - Julia: `SparseArrays.spdiagm`
  - Python: `scipy.sparse.spdiags`
- Matlab code:

```
dD = -min(mu, 0)/deta + sig.^2/(2*deta^2);
dM = min(mu, 0)/deta - max(mu, 0)/deta - sig.^2/deta^2;
dU = max(mu, 0)/deta + sig.^2/(2*deta^2);
M = spdiags([dD dM dU], [1 0 -1], N, N)';
```

Value Function Iteration

$$\rho v(\eta) = u(v, \eta) + \mu(v, \eta)\partial_\eta v(\eta) + \frac{1}{2}\sigma(v, \eta)^2 \partial_{\eta\eta} v(\eta)$$

$$\rho \mathbf{v} = \mathbf{u}(\mathbf{v}) + \mathbf{M}(\mathbf{v})\mathbf{v}$$

Value Function Iteration

$$\rho v(\eta) = u(v,\eta) + \mu(v,\eta)\partial_\eta v(\eta) + \frac{1}{2}\sigma(v,\eta)^2\partial_{\eta\eta} v(\eta)$$

$$\rho \mathbf{v} = \mathbf{u}(\mathbf{v}) + \mathbf{M}(\mathbf{v})\mathbf{v}$$

- Introduce time:

$$\rho v_t(\eta) = \partial_t v_t(\eta) + u(v_t,\eta) + \mu(v_t,\eta)\partial_\eta v_t(\eta) + \frac{1}{2}\sigma(v_t,\eta)^2\partial_{\eta\eta} v_t(\eta)$$

$$\partial_t v(t,\eta) = g(v,\eta)$$

- Explicit or Implicit?

Value Function Iteration

$$\rho v_t(\eta) = \partial_t v_t(\eta) + u(v_t, \eta) + \mu(v_t, \eta)\partial_\eta v_t(\eta) + \frac{1}{2}\sigma(v_t, \eta)^2 \partial_{\eta\eta} v_t(\eta)$$

- Explicit:

$$\rho \mathbf{v}_t = \frac{\mathbf{v}_t - \mathbf{v}_{t-\Delta t}}{\Delta t} + \mathbf{u}(\mathbf{v}_t) + \mathbf{M}(\mathbf{v}_t)\mathbf{v}_t$$

- Implicit:

$$\rho \mathbf{v}_{t-\Delta t} = \frac{\mathbf{v}_t - \mathbf{v}_{t-\Delta t}}{\Delta t} + \mathbf{u}(\mathbf{v}_{t-\Delta t}) + \mathbf{M}(\mathbf{v}_{t-\Delta t})\mathbf{v}_{t-\Delta t}$$

- We use:

$$\rho \mathbf{v}_{t-\Delta t} = \frac{\mathbf{v}_t - \mathbf{v}_{t-\Delta t}}{\Delta t} + \mathbf{u}(\mathbf{v}_t) + \mathbf{M}(\mathbf{v}_t)\mathbf{v}_{t-\Delta t}$$

$$\mathbf{v}_{t-\Delta t} = \Big((1 + \rho\Delta t)\mathbf{I} - \Delta t \mathbf{M}(\mathbf{v}_t)\Big)^{-1}\Big(\Delta t \mathbf{u}(\mathbf{v_t}) + \mathbf{v}_t\Big)$$

Alternative: Policy Function Iteration

$$\rho v(\eta) = u(v,\eta) + \mu(v,\eta)\partial_\eta v(\eta) + \frac{1}{2}\sigma(v,\eta)^2\partial_{\eta\eta} v(\eta)$$

$$\rho \mathbf{v} = \mathbf{u}(\mathbf{v}) + \mathbf{M}(\mathbf{v})\mathbf{v}$$

- Assume constant 'policy' along the entire path:
    1. Given $\mathbf{v}_k$, compute $\mathbf{u}(\mathbf{v_k}), \mathbf{M}(\mathbf{v}_k)$
    2. Update $\mathbf{v}_{k+1} = \Big(\rho\mathbf{I} - \mathbf{M}(\mathbf{v}_k)\Big)^{-1}\mathbf{u}(\mathbf{v_k})$

Kolmogorov Forward Equation

$$\partial_t g_t(\eta) = -\partial_\eta(\mu(\eta)g_t(\eta)) + \frac{1}{2}\partial_{\eta\eta}(\sigma(\eta)^2 g_t(\eta))$$

- Stationary version:

$$0 = -\partial_\eta(\mu(\eta)g(\eta)) + \frac{1}{2}\partial_{\eta\eta}(\sigma(\eta)^2 g(\eta))$$

- Discretize:

$$0 = \tilde{\mathbf{M}}\mathbf{g}$$

Kolmogorov Forward Equation

$$\partial_t g_t(\eta) = -\partial_\eta(\mu(\eta) g_t(\eta)) + \frac{1}{2}\partial_{\eta\eta}(\sigma(\eta)^2 g_t(\eta))$$

- Stationary version:

$$0 = -\partial_\eta(\mu(\eta) g(\eta)) + \frac{1}{2}\partial_{\eta\eta}(\sigma(\eta)^2 g(\eta))$$

- Discretize:

$$0 = \tilde{\mathbf{M}}\mathbf{g}$$

- Turns out $\tilde{\mathbf{M}} = \mathbf{M}'$!

Kolmogorov Forward Equation

- Solving for $\mathbf{g}$:
  1. Iterate

$$\frac{\mathbf{g}_{t+\Delta t} - \mathbf{g}_t}{\Delta t} = \mathbf{M}'\mathbf{g}_{t+\Delta t}$$

$$\mathbf{g}_{t+\Delta t} = \left(\mathbf{I} - \Delta t \mathbf{M}'\right)^{-1} \mathbf{g}_t$$

  2. Solve given g_1

$$\begin{bmatrix} M'_{11} & M'_{1,2:N} \\ M'_{2:N,1} & M'_{2:N,2:N} \end{bmatrix} \begin{bmatrix} g_1 \\ g_{2:N} \end{bmatrix} = 0 \iff g_{2:N} = -(M'_{2:N,2:N})^{-1} M'_{2:N,1} g_1$$

  3. Find the null space (kernel) of $\mathbf{M}'$ using some built-in method (Matlab: `null`)

Simple valuation problem

$$v(x, \sigma) = E\left[\int_0^\infty e^{-\rho t} u(x_t) dt \,\middle|\, x_0 = x, \sigma_0 = \sigma\right]$$

$$dx_t = -x_t dt + \sigma_t (1 - x_t^2) dZ_t$$

$$\sigma_t \in \{\sigma_L, \sigma_H\}$$

- σ changes with Poisson intensity λ

Simple valuation problem

$$v(x,\sigma) = E\left[\int_0^\infty e^{-\rho t} u(x_t) dt \;\middle|\; x_0 = x, \sigma_0 = \sigma\right]$$

$$dx_t = -x_t dt + \sigma_t (1 - x_t^2) dZ_t$$

$$\sigma_t \in \{\sigma_L, \sigma_H\}$$

- σ changes with Poisson intensity λ

$$\rho v(x,\sigma_L) = u(x) - x\partial_x v(x,\sigma_L) + \frac{\sigma_L^2 (1-x^2)^2}{2} \partial_{xx} v(x,\sigma_L) + \lambda(v(x,\sigma_H) - v(x,\sigma_L))$$

$$\rho v(x,\sigma_H) = u(x) - x\partial_x v(x,\sigma_H) + \frac{\sigma_H^2 (1-x^2)^2}{2} \partial_{xx} v(x,\sigma_H) + \lambda(v(x,\sigma_L) - v(x,\sigma_H))$$

Simple valuation problem

- Fix $\sigma = \sigma_L$ and discretize $x \in \{x_1, \ldots, x_N\}$:

$$\rho v(x_i, \sigma_L) = u(x_i) - x_i \partial_x v(x_i, \sigma_L) + \frac{\sigma_L^2 (1 - x_i^2)^2}{2} \partial_{xx} v(x_i, \sigma_L) + \lambda (v(x_i, \sigma_H) - v(x_i, \sigma_L))$$

Simple valuation problem

- Fix $\sigma = \sigma_L$ and discretize $x \in \{x_1, \ldots, x_N\}$:

$$\rho v(x_i, \sigma_L) = u(x_i) - x_i \partial_x v(x_i, \sigma_L) + \frac{\sigma_L^2 (1 - x_i^2)^2}{2} \partial_{xx} v(x_i, \sigma_L) + \lambda (v(x_i, \sigma_H) - v(x_i, \sigma_L))$$

- Given σ, construct M_{σ_L} (NxN) and M_{σ_H} (NxN)
- Stack states together: $\{\sigma_L, x_1\}, \ldots, \{\sigma_L, x_N\}, \{\sigma_H, x_1\}, \ldots, \{\sigma_H, x_N\}$:

$$M = \begin{array}{cc} & \begin{array}{cc} \sigma_L & \sigma_H \end{array} \\ \begin{array}{c} \sigma_L \\ \sigma_H \end{array} & \begin{bmatrix} M_{\sigma_L} & 0 \\ 0 & M_{\sigma_H} \end{bmatrix} \end{array} + \Lambda$$

Simple valuation problem

- Constructing Λ:
  1. Transition matrix between σ-s:

$$\Lambda_0 = \begin{bmatrix} -\lambda & \lambda \\ \lambda & -\lambda \end{bmatrix}$$

  2. Use Kronecker product ($\texttt{kron}(\Lambda_0, \mathbf{I}_N)$):

$$\Lambda = \begin{bmatrix} -\lambda \mathbf{I}_N & \lambda \mathbf{I}_N \\ \lambda \mathbf{I}_N & -\lambda \mathbf{I}_N \end{bmatrix}$$

Simple valuation problem

$$M = \begin{array}{cc} & \begin{matrix} & & \sigma_L & & & & & \sigma_H & & \end{matrix} \\ \begin{matrix} \\ \\ \sigma_L \\ \\ \\ \\ \\ \sigma_H \\ \\ \\ \end{matrix} & \begin{bmatrix} - & + & 0 & 0 & 0 & + & 0 & 0 & 0 & 0 \\ + & - & + & 0 & 0 & 0 & + & 0 & 0 & 0 \\ 0 & + & - & + & 0 & 0 & 0 & + & 0 & 0 \\ 0 & 0 & + & - & + & 0 & 0 & 0 & + & 0 \\ 0 & 0 & 0 & + & - & 0 & 0 & 0 & 0 & + \\ + & 0 & 0 & 0 & 0 & - & + & 0 & 0 & 0 \\ 0 & + & 0 & 0 & 0 & + & - & + & 0 & 0 \\ 0 & 0 & + & 0 & 0 & 0 & + & - & + & 0 \\ 0 & 0 & 0 & + & 0 & 0 & 0 & + & - & + \\ 0 & 0 & 0 & 0 & + & 0 & 0 & 0 & + & - \end{bmatrix} \end{array}$$