On the Recursive Saddle Point Method

Messner and Pavoni

April 29, 2008

Introduction

- An example on the MM(98) method that

  1. admits solutions that may not be correct

  2. delivers though the correct value of the program

Overview of the MM

- Optimization problems with forward-looking constraints (Euler equations, IC constraints, etc).

- *MM*: turn a *sequential* saddle point to a *recursive* saddle point: use as a state variable the *multipliers* on the constraints.

Example of Messner and Pavoni

Principal-agent

$$\sup_{a_t} \sum_{t=0}^{\infty} \beta^t (y - a_t)$$

s.t.

$$\begin{aligned} a_t &\in [0, \bar{a}], t \geq 0 \\ \sum_{i=0}^{\infty} \beta^i u(a_{t+i}) &\geq \frac{b}{1-\beta}, \forall t \geq 0 \end{aligned}$$

- linear utility: $u(a) = a$

Sequential Saddle Point

Define Lagrangian

$$L(a,\lambda) = \sum_{t=0}^{\infty} \beta^t \left(y - a_t + \lambda_t \left[\sum_{i=0}^{\infty} \beta^i a_{t+i} - \frac{b}{1-\beta} \right] \right)$$

Definition: (a^*, λ^*) is a saddle-point of L if:

$$L(a,\lambda^*) \leq L(a^*,\lambda^*) \leq L(a^*,\lambda) \forall \lambda \geq 0, a_t \in [0, \bar{a}]$$

$$L(a^*,\lambda^*) = \inf_{\lambda \geq 0} \sup_{a_t \in [0,\bar{a}]} L(a,\lambda) = \sup_{a_t \in [0,\bar{a}]} \inf_{\lambda \geq 0} L(a,\lambda)$$

Solution to the problem:

$$\begin{aligned} a_t &\in [0, \bar{a}] \\ \sum_{t=0}^{\infty} \beta^t a_t &= \frac{b}{1-\beta} \\ \sum_{t=0}^{t-1} \beta^i [b - a_i] &\geq \beta^t (a_t - b) \\ \lambda_0 &= 1, \lambda_t = 0, t \geq 1 \end{aligned}$$

- Multiplicity of solutions and not only $a_t = b$.

MM (not exactly like Messner/Pavoni)

Rewrite the Lagrangian as

$$\begin{aligned} L &= \sum_{t=0}^{\infty} \beta^t (y - a_t + a_t \underbrace{\sum_{i=0}^{t} \lambda_t}_{\text{past promises}} - \lambda_t \frac{b}{1-\beta}) \\ &= \sum_{t=0}^{\infty} \beta^t (y - a_t + a_t \psi_t - \lambda_t \frac{b}{1-\beta}) \end{aligned}$$

where

$$\psi_t = \psi_{t-1} + \lambda_t, \psi_{-1} \equiv 0$$

Sequential Saddle point $\longrightarrow$*Recursive Saddle point*

$$W(\psi_{-}) = \inf_{\lambda \geq 0} \sup_{a \in [0, \bar{a}]} \{y + (\psi - 1)a - \lambda \frac{b}{1-\beta} + \beta W(\psi)\}$$

s.t.

$$\psi = \psi_{-} + \lambda$$

- Policy functions: $a_t = a(\psi_{t-1}), \lambda_t = \lambda(\psi_{t-1})$.
- Value of the program $W(0)$

- Value function *linear*

$$W(\psi_{t-1}) = \left\{ \begin{array}{ll} \frac{y-b}{1-\beta} + \frac{b}{1-\beta}\psi_{t-1} & , \psi_{t-1} \leq 1 \\ \\ \frac{y-\bar{a}}{1-\beta} + \frac{\bar{a}}{1-\beta}\psi_{t-1} & , \psi_{t-1} > 1 \end{array} \right\}$$

- Policy function

$$\begin{aligned} \lambda_t &= \left\{ \begin{array}{ll} 1 - \psi_{t-1} & , \psi_{t-1} \leq 1 \\ 0 & , \psi_{t-1} > 1 \end{array} \right\} \\ a_t &= \left\{ \begin{array}{ll} b & \psi_{t-1} < 1 \\ \bar{a} & \psi_{t-1} > 1 \end{array} \right\} \end{aligned}$$

- Find *correct* value of program, *correct* multipliers but *Problem* with the conditions for the policy function a.

- Messner/Pavoni get a *superset* of the right conditions (I get a *subset*; if correct...).

- *In any case:* the intertemporal link of the no default conditions is not imposed. there is no intertemporal link

- The no-default-condition is imposed only *implicitly* through the *multipliers*.

Where are the constraints hidden??

- foc wrt $\lambda_t \geq 0$

$$a_t + \beta W_\psi(\psi_t) \geq \frac{b}{1-\beta}$$

- Envelope:

$$W_\psi(\psi_{t-1}) = a_t + \beta W_\psi(\psi_t)$$

- The envelope condition (if valid) would enforce the constraint.

Keep continuation utility as state variable

$$V(U) = \sup_{a \in [0,\bar{a}], U'} y - a + \beta V(U')$$

s.t.

$$\begin{aligned} a + \beta U' &\geq \frac{b}{1-\beta} \\ U &= a + \beta U' \end{aligned}$$

- initial value of the program $V(\frac{b}{1-\beta})$

- Source of the problem: multiplier may not be a *"sharp"* characterization of the state.

- There may not be a *"1-1"* relationship between multipliers (shadow values) and continuation utilities.

- So there may be *information lost* during the step from the primal variable (U) to the dual (ψ_t)

- Messner and Pavoni think that this may not be a problem if the problem was *strictly* concave.

Potential Remedies

- In applications we usually do not have concave problems.
- The existence of a sequential saddle point is not guarranted (i.e. first-order conditions not sufficient)
- However if the second-order conditions for a maximum were satisfied we could have a *local* saddle point and a locally *strictly* concave program and then it may be ok...